

МОСКОВСКИЙ ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ
(ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ)

На правах рукописи

УДК 519.254

Исаченко Роман Владимирович

ВЫБОР МОДЕЛИ ДЕКОДИРОВАНИЯ СИГНАЛОВ В ПРОСТРАНСТВАХ
ВЫСОКОЙ РАЗМЕРНОСТИ

05.13.17 — Теоретические основы информатики

Диссертация на соискание ученой степени
кандидата физико-математических наук

Научный руководитель:
д.ф.-м.н. В. В. Стрижов

Москва — 2019

Оглавление

	Стр.
Введение	4
Глава 1. Постановка задачи декодирования	8
1.1. Задача декодирования	8
1.2. Временные ряды	9
1.3. Линейная регрессия	11
1.4. Методы снижения размерности пространства	12
1.4.1. Метод Главных Компонент (PCA)	12
1.4.2. PLS	12
1.4.3. CCA	18
1.4.4. High order PLS/CCA	20
1.4.5. Многовидовые данные	20
1.4.6. Deep CCA	26
1.5. Вычислительный эксперимент	27
Глава 2. Выбор признаков	31
2.1. Выбор признаков с помощью квадратичного программирования . . .	33
2.2. Многоиндексный метод выбора признаков	34
2.2.1. Агрегация релевантностей целевых переменных	34
2.2.2. Симметричный учёт значимости признаков и целевых переменных	36
2.2.3. Минимаксная постановка задачи выбора признаков	37
2.2.4. Несимметричный учёт значимостей признаков и целевых переменных	40
2.3. Вычислительный эксперимент	42
Глава 3. Выбор параметров нелинейных моделей	48
3.1. Выбор параметров для обучения моделей	48
3.2. Метод Ньютона решения задачи настройки параметров	50

3.3. Метод Ньютона с выбором параметром с помощью квадратичного программирования	53
3.4. Вычислительный эксперимент	53
Глава 4. Метрические методы	
4.1. Метрическое обучение в задачах кластеризации временных рядов . .	58
4.2. Алгоритм адаптивного метрического обучения	60
4.3. Постановка задачи	63
4.3.1. Выравнивание временных рядов.	63
4.3.2. Метрическое обучение.	65
4.3.3. Классификация временных рядов.	66
4.4. Вычислительный эксперимент	67
4.5. Вычислительный эксперимент	68
Глава 5. Порождение признаков с помощью метамоделей	
5.1. Постановка задачи	75
5.2. Порождение признаков	76
5.3. Классификация временных рядов	79
5.4. Вычислительный эксперимент	81
Глава 6. Анализ прикладных задач	
Введение	88

Введение

Диссертационная работа посвящена построению математических моделей машинного обучения в пространствах высокой размерности. Разработанные методы позволяют учесть зависимости, имеющиеся в исходных данных, с целью построения простой и устойчивой модели.

Актуальность темы.

В работе исследуется задача декодирования сигналов. При построении моделей машинного обучения возникает необходимость построения низкоразмерного признакового пространства. Требуется по входному исходному сигналу предсказать отклик на этот сигнал.

Сложностью задачи является избыточность исходного описания данных. Исходное признаковое пространство является мультикоррелированным. Финальная предсказательная модель оказывается неустойчивой. Для построения простой, устойчивой модели применяются методы снижения размерности пространства [1, 2] и выбора признаков [3, 4].

В работе рассматриваются задачи с векторной целевой переменной. При предсказании векторной целевой переменной возникает необходимости в анализе структуры целевого пространства. Целевое пространство содержит зависимости. В работе предлагаются методы, которые позволяют учесть зависимости как в исходном пространстве объектов, так и в пространстве целевой переменной.

Методы снижения размерности пространства позволяют снизить размерность исходного пространства объектов, сложность модели существенно снижается. Алгоритмы снижения размерности позволяют найти оптимальные комбинации исходных признаков. В случае если количество таких комбинаций существенно меньше, чем число исходных признаков, то полученное представление снижает размерность. Цель снижения размерности найти наиболее репрезентативные и информативные комбинации признаков для решения целевой задачи.

Выбор признаков является частным случаем снижения размерности пространства. Найденные комбинации признаков являются лишь подмножеством исходных признаков. Таким образом появляется возможность отсеять шумовые неинформативные признаки. Процедура выбора признаков может как зависеть, так и не зависеть от модели предсказания.

После нахождения оптимального представления данных с помощью снижения размерности, возникает задача нахождения правильной метрики в скрытом пространстве объектов. В случае евклидова пространства естественным выбором метрики оказывается квадратичная норма. Задача метрического обучения найти оптимальную метрику, связывающую объекты.

Цели работы.

1. Исследовать задачу декодирования сигналов с многомерной целевой переменной.
2. Предложить способ снижения размерности пространства, учитывающий зависимости в исходном пространстве сигналов, а также в целевом пространстве.
3. Предложить процедуру выбора признаков для задачи декодирования сигналов.
4. Исследовать линейные и нелинейные модели для решения поставленной модели, получить теоретические оценки оптимальности моделей.
5. Провести вычислительный эксперимент для проверки адекватности предложенных методов.

Основные положения, выносимые на защиту.

1. Метод снижения размерности пространства, отображающий независимую и целевую переменные в единое скрытое низкоразмерное представление.
2. Методы выбора признаков для задач с многомерной целевой переменной, учитывающие структуры пространств.

3. Алгоритм выбора наиболее влиятельных параметров для оптимизации нелинейной модели.
4. Алгоритм метрического обучения для временных рядов с процедурой их выравнивания.
5. Программный комплекс, включающий прогностические модели для высокоразмерных данных. Проведены вычислительные эксперименты, подтверждающие адекватность методов.

Методы исследования. Для достижения поставленных целей используются линейные и нелинейные алгоритмы регрессии. Для анализа временных рядов используются классические авторегрессионные методы. Для извлечения признаков используются частотные характеристики временного ряда. Для построения скрытого пространства используются линейные методы снижения размерности пространства, их нелинейные модификации, а также нейросетевые методы. Для выбора признаков наряду с классическими методами, используются методы, основанные на решении задачи квадратичного программирования. Для построения метрического пространства используются методы условной выпуклой оптимизации.

Научная новизна.

Теоретическая значимость.

Практическая значимость.

Степень достоверности и апробация работы. Достоверность результатов подтверждена математическими доказательствами, экспериментальной проверкой результатов пределаемых алгоритмов на реальных данных, публикациями результатов в рецензируемых научных изданиях, в том числе рекомендованных ВАК. Результаты работы докладывались и обсуждались на следующих научных конференциях.

1. Международная научная конференция «Ломоносов», 2016, [5].
2. Международная научная конференция «11th International Conference on Intelligent Data Processing: Theory and Applications», 2016, [6].
3. Всероссийская научная конференция «Математические методы распознавания образов», 2017, [7].
4. Международная научная конференция «12th International Conference on Intelligent Data Processing: Theory and Applications», 2018, [8].
5. Международная научная конференция «13th International Conference on Intelligent Data Processing: Theory and Applications», 2020, ???.

Работа поддержана грантами Российского фонда фундаментальных исследований.

1. ???

Публикации по теме диссертации.

Структура и объем работы.

Личный вклад. Все приведенные результаты, кроме отдельно оговоренных случаев, получены диссертантом лично при научном руководстве д.ф.-м.н. В. В. Стрижова.

Краткое содержание работы по главам.

Глава 1

Постановка задачи декодирования

1.1. Задача декодирования

Задача декодирования формулируется следующим образом. Требуется построить регрессионную модель между объектами из двух пространств: \mathbb{X} и \mathbb{Y} . Пространства обладают высокой, избыточной размерностью. Регрессионная модель оказывается неустойчивой. Для построения простой, точной и устойчивой модели предлагается учесть наличие внутренней структуры обоих пространств. Применяются методы снижения размерности пространств для независимой и целевой переменной. Итоговая регрессионная модель строится путём согласования образов исходных объектов в низкоразмерном пространстве.

Формализуем описанную задачу. Пусть $\mathbf{x} \in \mathbb{X} \subset \mathbb{R}^n$ – независимая переменная, $\mathbf{y} \in \mathbb{Y} \subset \mathbb{R}^r$ – целевая переменная.

Определение 1. Назовём *примером* пару (\mathbf{x}, \mathbf{y}) , состоящую из реализации независимой переменной $\mathbf{x} \in \mathbb{X}$ и целевой переменной $\mathbf{y} \in \mathbb{Y}$.

Определение 2. *Выборкой* $\mathcal{D} = (\mathbf{X}, \mathbf{Y})$ будем называть заданное множество примеров $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m$. Здесь $\mathbf{X} \in \mathbb{R}^{m \times n}$ – матрица независимой переменной, $\mathbf{Y} \in \mathbb{R}^{n \times k}$ – матрица целевой переменной:

$$\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_m]^\top = [\boldsymbol{\chi}_1, \dots, \boldsymbol{\chi}_n]; \quad \mathbf{Y} = [\mathbf{y}_1, \dots, \mathbf{y}_m]^\top = [\boldsymbol{\nu}_1, \dots, \boldsymbol{\nu}_r].$$

Столбцы $\boldsymbol{\chi}_j, j = 1, \dots, n$ матрицы \mathbf{X} являются признаками объекта, столбцы $\boldsymbol{\nu}_j, j = 1, \dots, r$ матрицы \mathbf{Y} являются целевыми столбцами.

Определение 3. Назовём *моделью* отображение $f : \mathbb{X} \rightarrow \mathbb{Y}$ из пространства независимой переменной в пространство целевой переменной.

Задача восстановления регрессии состоит в нахождении оптимальной модели f^* по известной выборке \mathcal{D} . Под оптимальностью понимается нахождение такой модели, которая бы доставляла минимум некоторой функции ошибки

$$f^* = \arg \min_f \mathcal{L}(f, \mathbf{X}, \mathbf{Y}). \tag{1.1}$$

Задача поиска оптимальной модели является задачей функциональной оптимизации. Для сужения пространства поиска моделей будем рассматривать параметрические модели $f(\mathbf{x}, \Theta)$, где Θ являются *параметрами модели*. Тогда задача (1.1) сводится к задаче поиска набора оптимальных параметров

$$\Theta^* = \arg \min_{\Theta} \mathcal{L}(\Theta, \mathbf{X}, \mathbf{Y}). \quad (1.2)$$

В данной диссертации рассматривается случай, когда размерность пространств \mathbb{X}, \mathbb{Y} является избыточной. В таком случае решение задачи (1.2) оказывается неустойчивым. Рассмотрим в качестве примера задачу восстановления линейной регрессии.

1.2. Временные ряды

Рассмотрим задачу декодирования временных рядов.

Определение 4. Пусть $\mathbf{S} = [\mathbf{s}_1, \dots, \mathbf{s}_T]$ – *сегмент многомерного временного ряда*. Вектор \mathbf{s}_t содержит значения временного ряда в момент времени t . В случае одномерного временного ряда \mathbf{s}_t является скаляром.

Определение 5. Назовём *задачей декодирования временных рядов* задачу восстановления значений сегментов целевых временных рядов \mathbf{S}_y по сегментам наблюдаемых независимых временных рядов \mathbf{S}_x .

Если сегмент целевого временного ряда \mathbf{S}_y является продолжением наблюдаемого сегмента \mathbf{S}_x задача декодирования сводится к построению прогностическом модели временного ряда. Для построения модели прогнозирования временных рядов широко используются два класса линейных методов: авторегрессионные модели [9, 10] и модели скользящего среднего [9, 10]. Авторегрессионные модели AR(p) строят прогноз в виде линейной комбинации p предыдущих значений временного ряда. Модели скользящего среднего MA(q) вместо предыдущих значений временного ряда используют комбинацию ошибок. Модель ARMA(p, q) [11] является комбинацией двух описанных подходов. ARMA($p,$

q) задает модель как линейную комбинацию p предыдущих значений временного ряда и q предыдущих значений ошибок. Для нахождения оптимальных параметров p и q модели ARMA используются автокорреляционная и частная автокорреляционная функции.

Модель ARMA используется для стационарных временных рядов, отвечающим строгим статистическим предположениям. На практике встречается огромное количество нестационарных временных рядов подверженных тренду, сезонности или цикличности. Модель ARIMA(p, d, q) [11] обобщает модель ARMA для случая нестационарных временных рядов. ARIMA берёт разности порядка d от исходного временного ряда для достижения стационарности данных. При этом на практике оказывается достаточным положить $d = 1$. Заметим, что при $d = 0$ модель ARIMA эквивалентна модели ARMA. Полезным обобщение модели ARIMA является модель AFRIMA [12]. Модель позволяет задать параметр d в виде вещественного числа.

Модель ARIMA плохо справляется с сезонными временными рядами. В работе [9] была предложена модель SARIMA, которая вводит в модель учет сезонной компоненты.

Задача декодирования временных рядов декомпозируется на следующие подзадачи.

- Порождение признакового пространства. Данный этап включает в себя процедуру извлечения признаков из исходных значений сигналов. Процедура порождения признакового пространства может быть основана на экспертных знания или же являться моделью машинного обучения. Данная подзадача подробно рассмотрена в главе ???.
- Снижение размерности пространства или выбор признаков. Исходные временные ряды, а также порожденное признаковое пространство оказываются избыточным, что приводит к избыточности и неустойчивости модели. Алгоритмы снижения размерности и выбора признаков подробно изложены в главе ???.

- Построение модели.

1.3. Линейная регрессия

Предполагается, что между объектами \mathbf{x} и ответами \mathbf{y} существует зависимость вида

$$\mathbf{y} = f(\mathbf{x}, \Theta) + \varepsilon, \quad (1.3)$$

где f – параметрическая модель регрессионной зависимости, Θ – пространство параметров модели, $\varepsilon \in \mathbb{R}^m$ – вектор регрессивных остатков. Необходимо восстановить зависимость f по заданной наблюдаемой выборке \mathcal{D} .

Предположим, что зависимость $f(\mathbf{x}, \Theta)$ линейная:

$$\mathbf{y} = f(\mathbf{x}, \Theta) + \varepsilon = \Theta \mathbf{x} + \varepsilon, \quad (1.4)$$

где $\Theta \in \mathbb{R}^{r \times n}$ – матрица параметров модели.

Необходимо найти матрицу параметров модели Θ при известной выборке $\mathcal{D} = (\mathbf{X}, \mathbf{Y})$. Оптимальные параметры Θ определяются минимизацией функции ошибки $\mathcal{L}(\Theta, \mathbf{X}, \mathbf{Y})$. При решении задачи линейной регрессии в качестве такой функции ошибки рассматривается квадратичная функция потерь:

$$\mathcal{L}(\Theta, \mathbf{X}, \mathbf{Y}) = \left\| \mathbf{Y}_{m \times r} - \mathbf{X}_{m \times n} \cdot \Theta_{n \times r}^\top \right\|_2^2 \rightarrow \min_{\Theta}. \quad (1.5)$$

Решением (1.5) является следующая матрица:

$$\Theta = \mathbf{Y}^\top \mathbf{X} (\mathbf{X}^\top \mathbf{X})^{-1}.$$

Наличие линейной зависимости между столбцами матрицы \mathbf{X} приводит к неустойчивому решению задачи оптимизации (1.5). Если существует вектор $\alpha \neq \mathbf{0}_n$ такой, что $\mathbf{X}\alpha = \mathbf{0}_m$, то добавление α к любому столбцу матрицы Θ не меняет значение функции потерь $\mathcal{L}(\Theta, \mathbf{X}, \mathbf{Y})$. В этом случае матрица $\mathbf{X}^\top \mathbf{X}$ близка к сингулярной и не обратима. Чтобы избежать сильной линейной зависимости между признаками, в данной работе исследуются методы снижения размерности и выбора признаков.

1.4. Методы снижения размерности пространства

1.4.1. Метод Главных Компонент (PCA)

Для устранения линейной зависимости и снижения размерности входного пространства объектов широко используется метод главных компонент (principal component analysis, PCA). Метод PCA находит низкоразмерное представление матрицы $\mathbf{X} = \mathbf{T}\mathbf{P}^T$, такое что новое представление $\mathbf{T} \in \mathbb{R}^{m \times l}$ содержит максимальную долю дисперсии исходной матрицы. При этом матрица отображения $\mathbf{P} \in \mathbb{R}^{n \times l}$ является ортогональной ($\mathbf{P}^T\mathbf{P} = \mathbf{I}$) и содержит правые собственные вектора матрицы ковариаций $\mathbf{X}^T\mathbf{X}$.

Метод PCA является базовым методом снижения размерности пространства. Существует множество модификаций базового метода. Вероятностный PCA [13] рассматривает задачу снижения размерности в терминах вероятностной модели, решая задачу с помощью вариационного EM алгоритма. Разреженный PCA [14] вводит в постановку задачи lasso регуляризацию для того, чтобы сделать матрицу отображения \mathbf{P} разреженной и более интерпретируемой. Нелинейный ядерный PCA [15] отображает исходные данные с помощью нелинейного отображения и использует RKHS для решения исходной задачи.

После нахождения матрицы отображения \mathbf{P} задача (1.5) принимает вид

$$\mathcal{L}(\mathbf{B}, \mathbf{T}, \mathbf{Y}) = \left\| \mathbf{Y}_{m \times r} - \mathbf{T}_{m \times l} \cdot \mathbf{B}_{r \times l}^T \right\|_2^2 \rightarrow \min_{\mathbf{B}}. \quad (1.6)$$

Модель прогнозирования (1.4) в случае снижения размерности с помощью PCA принимает вид:

$$\mathbf{y} = \mathbf{B}\mathbf{t} + \boldsymbol{\varepsilon} = \mathbf{B}\mathbf{P}\mathbf{x} + \boldsymbol{\varepsilon} = \boldsymbol{\Theta}\mathbf{x} + \boldsymbol{\varepsilon}, \text{ где } \boldsymbol{\Theta} = \mathbf{B}\mathbf{P}. \quad (1.7)$$

1.4.2. PLS

Основным недостатком метода PCA является отсутствие учёта взаимосвязи между признаками \mathbf{x}_j и целевыми векторами $\boldsymbol{\nu}_j$. Алгоритм частичных наи-

меньших квадратов проецирует матрицу объектов \mathbf{X} и матрицу ответов \mathbf{Y} в скрытое пространство малой размерностью l ($l < n$). Алгоритм PLS находит в скрытом пространстве матрицы $\mathbf{T}, \mathbf{U} \in \mathbb{R}^{m \times l}$, которые лучше всего описывают оригинальные матрицы \mathbf{X} и \mathbf{Y} . При этом PLS максимизирует ковариацию между столбцами \mathbf{t} и \mathbf{u} матриц \mathbf{T} и \mathbf{U} соответственно.

Алгоритм PLS был впервые предложен в работах [16, 17, 18]. Подробное описание алгоритма приведено в работах [19, 20, 21, 22, 23]. В работах [24, 25] приведен обзор обобщений базовой модели PLS. В работе [1] приведена модификация алгоритма PLS для получения разреженного набора признаков.

Матрица объектов \mathbf{X} и целевая матрица \mathbf{Y} проецируются на латентное пространство следующим образом:

$$\underset{m \times n}{\mathbf{X}} = \underset{m \times l}{\mathbf{T}} \cdot \underset{l \times n}{\mathbf{P}^\top} + \underset{m \times n}{\mathbf{F}} = \sum_{k=1}^l \underset{m \times 1}{\mathbf{t}_k} \cdot \underset{1 \times n}{\mathbf{p}_k^\top} + \underset{m \times n}{\mathbf{F}}, \quad (1.8)$$

$$\underset{m \times r}{\mathbf{Y}} = \underset{m \times l}{\mathbf{U}} \cdot \underset{l \times r}{\mathbf{Q}^\top} + \underset{m \times r}{\mathbf{E}} = \sum_{k=1}^l \underset{m \times 1}{\mathbf{u}_k} \cdot \underset{1 \times r}{\mathbf{q}_k^\top} + \underset{m \times r}{\mathbf{E}}. \quad (1.9)$$

Здесь \mathbf{T} и \mathbf{U} – образы исходных матриц в скрытом пространстве, причём столбцы матрицы \mathbf{T} ортогональны; \mathbf{P} и \mathbf{Q} – матрицы перехода; \mathbf{E} и \mathbf{F} – матрицы остатков. Алгоритм PLS максимизирует линейную зависимость между столбцами матриц \mathbf{T} и \mathbf{U}

$$\mathbf{U} \approx \mathbf{T}\mathbf{B}, \quad \mathbf{B} = \text{diag}(\beta_k), \quad \beta_k = \mathbf{u}_k^\top \mathbf{t}_k / (\mathbf{t}_k^\top \mathbf{t}_k)$$

Алгоритм решает следующую оптимизационную задачу:

$$\max_{\|\mathbf{p}\|_2 = \|\mathbf{q}\|_2 = 1} [\text{cov}(\mathbf{X}\mathbf{p}, \mathbf{Y}\mathbf{q})^2] = \max_{\mathbf{p}, \mathbf{q}} \frac{\mathbf{p}^\top \mathbf{X}^\top \mathbf{Y}\mathbf{q}}{\sqrt{\mathbf{p}^\top \mathbf{p}} \sqrt{\mathbf{q}^\top \mathbf{q}}}. \quad (1.10)$$

Псевдокод метода регрессии PLS приведен в алгоритме 1. Алгоритм итеративно на каждом из l шагов вычисляет по одному столбцу $\mathbf{t}_k, \mathbf{u}_k, \mathbf{p}_k, \mathbf{q}_k$ матриц $\mathbf{T}, \mathbf{U}, \mathbf{P}, \mathbf{Q}$ соответственно. После вычисления следующего набора векторов из матриц \mathbf{X}, \mathbf{Y} вычитаются очередные одноранговые аппроксимации. При этом

предполагается, что исходные матрицы \mathbf{X} и \mathbf{Y} нормированы (имеют нулевое среднее и единичное среднее отклонение).

Algorithm 1 Алгоритм PLS

Вход: $\mathbf{X}, \mathbf{Y}, l$;

Выход: $\mathbf{T}, \mathbf{P}, \mathbf{Q}$;

- 1: нормировать матрицы \mathbf{X} и \mathbf{Y} по столбцам
 - 2: инициализировать \mathbf{u}_0 (первый столбец матрицы \mathbf{Y})
 - 3: $\mathbf{X}_1 = \mathbf{X}; \mathbf{Y}_1 = \mathbf{Y}$
 - 4: **для** $k = 1, \dots, l$
 - 5: **повторять**
 - 6: $\mathbf{w}_k := \mathbf{X}_k^\top \mathbf{u}_{k-1} / (\mathbf{u}_{k-1}^\top \mathbf{u}_{k-1}); \quad \mathbf{w}_k := \frac{\mathbf{w}_k}{\|\mathbf{w}_k\|}$
 - 7: $\mathbf{t}_k := \mathbf{X}_k \mathbf{w}_k$
 - 8: $\mathbf{c}_k := \mathbf{Y}_k^\top \mathbf{t}_k / (\mathbf{t}_k^\top \mathbf{t}_k); \quad \mathbf{c}_k := \frac{\mathbf{c}_k}{\|\mathbf{c}_k\|}$
 - 9: $\mathbf{u}_k := \mathbf{Y}_k \mathbf{c}_k$
 - 10: **пока** \mathbf{t}_k не стабилизируется
 - 11: $\mathbf{p}_k := \mathbf{X}_k^\top \mathbf{t}_k / (\mathbf{t}_k^\top \mathbf{t}_k), \mathbf{q}_k := \mathbf{Y}_k^\top \mathbf{t}_k / (\mathbf{t}_k^\top \mathbf{t}_k)$
 - 12: $\mathbf{X}_{k+1} := \mathbf{X}_k - \mathbf{t}_k \mathbf{p}_k^\top$
 - 13: $\mathbf{Y}_{k+1} := \mathbf{Y}_k - \mathbf{t}_k \mathbf{q}_k^\top$
-

Вектора \mathbf{t}_k и \mathbf{u}_k из внутреннего цикла алгоритма 1 содержат информацию о матрице объектов \mathbf{X} и матрице ответов \mathbf{Y} соответственно. Блоки из шагов (6)-(7) и шагов (8)-(9) — аналоги алгоритма PCA для матриц \mathbf{X} и \mathbf{Y} [26]. Последовательное выполнение блоков позволяет учесть взаимную связь между матрицами \mathbf{X} и \mathbf{Y} .

Теоретическое обоснование алгоритма PLS следует из следующих утверждений.

Утверждение 1. Максимизация ковариации между векторами \mathbf{t}_k и \mathbf{u}_k сохраняет дисперсию матриц \mathbf{X} и \mathbf{Y} и учитывает их линейную зависимость.

Доказательство. Утверждение следует из равенства

$$\text{cov}(\mathbf{t}_k, \mathbf{u}_k) = \text{corr}(\mathbf{t}_k, \mathbf{u}_k) \cdot \sqrt{\text{var}(\mathbf{t}_k)} \cdot \sqrt{\text{var}(\mathbf{u}_k)}.$$

Максимизация дисперсий векторов \mathbf{t}_k и \mathbf{u}_k отвечает за сохранение информации об исходных матрицах, корреляция между векторами отвечает взаимосвязи между \mathbf{X} и \mathbf{Y} . \square

Во внутреннем цикле алгоритма вычисляются нормированные вектора весов \mathbf{w}_k и \mathbf{c}_k . Из данных векторов строятся матрицы весов \mathbf{W} и \mathbf{C} соответственно.

Утверждение 2. В результате выполнения внутреннего цикла вектора \mathbf{w}_k и \mathbf{c}_k будут собственными векторами матриц $\mathbf{X}_k^\top \mathbf{Y}_k \mathbf{Y}_k^\top \mathbf{X}_k$ и $\mathbf{Y}_k^\top \mathbf{X}_k \mathbf{X}_k^\top \mathbf{Y}_k$, соответствующими максимальным собственным значениям.

$$\mathbf{w}_k \propto \mathbf{X}_k^\top \mathbf{u}_{k-1} \propto \mathbf{X}_k^\top \mathbf{Y}_k \mathbf{c}_{k-1} \propto \mathbf{X}_k^\top \mathbf{Y}_k \mathbf{Y}_k^\top \mathbf{t}_{k-1} \propto \mathbf{X}_k^\top \mathbf{Y}_k \mathbf{Y}_k^\top \mathbf{X}_k \mathbf{w}_{k-1},$$

$$\mathbf{c}_k \propto \mathbf{Y}_k^\top \mathbf{t}_k \propto \mathbf{Y}_k^\top \mathbf{X}_k \mathbf{w}_k \propto \mathbf{Y}_k^\top \mathbf{X}_k \mathbf{X}_k^\top \mathbf{u}_{k-1} \propto \mathbf{Y}_k^\top \mathbf{X}_k \mathbf{X}_k^\top \mathbf{Y}_k \mathbf{c}_{k-1},$$

где символ \propto означает равенство с точностью до мультипликативной константы.

Доказательство. Утверждение следует из того факта, что правила обновления векторов \mathbf{w}_k , \mathbf{c}_k совпадают с итерацией алгоритма поиска максимального собственного значения. Данный алгоритм основан на следующем факте.

Если матрица \mathbf{A} диагонализуема, \mathbf{x} — некоторый вектор, то

$$\lim_{k \rightarrow \infty} \mathbf{A}^k \mathbf{x} = \lambda_{\max}(\mathbf{A}) \cdot \mathbf{v}_{\max},$$

где $\lambda_{\max}(\mathbf{A})$ — максимальное собственное значение матрицы \mathbf{A} , \mathbf{v}_{\max} — собственный вектор матрицы \mathbf{A} , соответствующий $\lambda_{\max}(\mathbf{A})$. \square

Утверждение 3. Обновление векторов по шагам (6)–(9) алгоритма 1 соответствует максимизации ковариации между векторами \mathbf{t}_k и \mathbf{u}_k .

Доказательство. Максимальная ковариация между векторами \mathbf{t}_k и \mathbf{u}_k равна максимальному собственному значению матрицы $\mathbf{X}_k^\top \mathbf{Y}_k \mathbf{Y}_k^\top \mathbf{X}_k$:

$$\begin{aligned} \max_{\mathbf{t}_k, \mathbf{u}_k} \text{cov}(\mathbf{t}_k, \mathbf{u}_k)^2 &= \max_{\substack{\|\mathbf{w}_k\|=1 \\ \|\mathbf{c}_k\|=1}} \text{cov}(\mathbf{X}_k \mathbf{w}_k, \mathbf{Y}_k \mathbf{c}_k)^2 = \max_{\substack{\|\mathbf{w}_k\|=1 \\ \|\mathbf{c}_k\|=1}} \text{cov}\left(\mathbf{c}_k^\top \mathbf{Y}_k^\top \mathbf{X}_k \mathbf{w}_k\right)^2 = \\ &= \max_{\|\mathbf{w}_k\|=1} \text{cov}\left\| \mathbf{Y}_k^\top \mathbf{X}_k \mathbf{w}_k \right\|^2 = \max_{\|\mathbf{w}_k\|=1} \mathbf{w}_k^\top \mathbf{X}_k^\top \mathbf{Y}_k \mathbf{Y}_k^\top \mathbf{X}_k \mathbf{w}_k = \\ &= \lambda_{\max} \left(\mathbf{X}_k^\top \mathbf{Y}_k \mathbf{Y}_k^\top \mathbf{X}_k \right), \end{aligned}$$

где $\lambda_{\max}(\mathbf{A})$ — максимальное собственное значение матрицы \mathbf{A} . Применяя утверждение 2, получаем требуемое. \square

После завершения внутреннего цикла на шаге (11) вычисляются вектора \mathbf{p}_k , \mathbf{q}_k проецированием столбцов матриц \mathbf{X}_k и \mathbf{Y}_k на вектор \mathbf{t}_k . Для перехода на следующий шаг необходимо вычесть из матриц \mathbf{X}_k и \mathbf{Y}_k одноранговые аппроксимации $\mathbf{t}_k \mathbf{p}_k^\top$ и $\mathbf{t}_k \mathbf{q}_k^\top$

$$\mathbf{X}_{k+1} = \mathbf{X}_k - \mathbf{t}_k \mathbf{p}_k^\top = \mathbf{X} - \sum_k \mathbf{t}_k \mathbf{p}_k^\top,$$

$$\mathbf{Y}_{k+1} = \mathbf{Y}_k - \mathbf{t}_k \mathbf{q}_k^\top = \mathbf{Y} - \sum_k \mathbf{t}_k \mathbf{q}_k^\top.$$

При этом каждый следующий вектор \mathbf{t}_k оказывается ортогонален всем векторам \mathbf{t}_i , $i = 1, \dots, k$.

Для получения прогнозов модели и нахождения параметров модели домножим справа формулу (1.8) на матрицу \mathbf{W} . Строки матрицы невязок \mathbf{E} ортогональны столбцам матрицы \mathbf{W} , поэтому

$$\mathbf{XW} = \mathbf{TP}^\top \mathbf{W}.$$

Линейное преобразование между объектами в исходном и латентном пространстве имеет вид

$$\mathbf{T} = \mathbf{XW}^*, \tag{1.11}$$

где $\mathbf{W}^* = \mathbf{W}(\mathbf{P}^\top \mathbf{W})^{-1}$.

Матрица параметров модели 1.4 находится из уравнений (1.9), (1.11)

$$\mathbf{Y} = \mathbf{T}\mathbf{Q}^T + \mathbf{E} = \mathbf{X}\mathbf{W}^*\mathbf{Q}^T + \mathbf{E} = \mathbf{X}\Theta + \mathbf{E}.$$

Таким образом, параметры модели (1.4) равны

$$\Theta = \mathbf{W}(\mathbf{P}^T\mathbf{W})^{-1}\mathbf{Q}^T. \quad (1.12)$$

Финальная модель (1.4.2.) является линейной, низкоразмерной в скрытом пространстве. Это снижает избыточность данных и повышает стабильность модели.

Для демонстрации разницы между алгоритмами PCA, PLS был проведен модельный эксперимент для случая, когда размерности пространств объектов, ответов и латентного пространства равны 2 ($n = r = l = 2$). На Рис. 1.1 показаны результаты работы методов. Синими и зелёными точками изображены объекты \mathbf{x}_i и целевые переменные \mathbf{y}_i . Точки \mathbf{X} сгенерированы из нормального распределения с нулевым матожиданием. Точки \mathbf{Y} линейным образом зависят от второй главной компоненты pc_2 матрицы \mathbf{X} и не зависят от первой главной компоненты pc_1 . Красным контуром показаны линии уровня матриц ковариаций распределений. Чёрным изображены единичные окружности. Красные стрелки соответствуют главным компонентам матриц \mathbf{X} и \mathbf{Y} . Чёрные стрелки соответствуют векторам матриц \mathbf{W} и \mathbf{C} алгоритма PLS. Учёт взаимной связи между матрицами \mathbf{X} и \mathbf{Y} отклоняет вектора \mathbf{w}_k и \mathbf{c}_k от направления главных компонент.

При снижении размерности пространств до одного признака алгоритм PCA выберет первую главную компоненту pc_1 , отбросив компоненту pc_2 , так как первая компонента объясняет большую части дисперсии исходной матрицы \mathbf{X} . При этом матрица \mathbf{Y} не зависит от pc_1 . Тем самым финальная модель окажется не оптимальной. Алгоритм PLS позволяет побороться с данной проблемой.

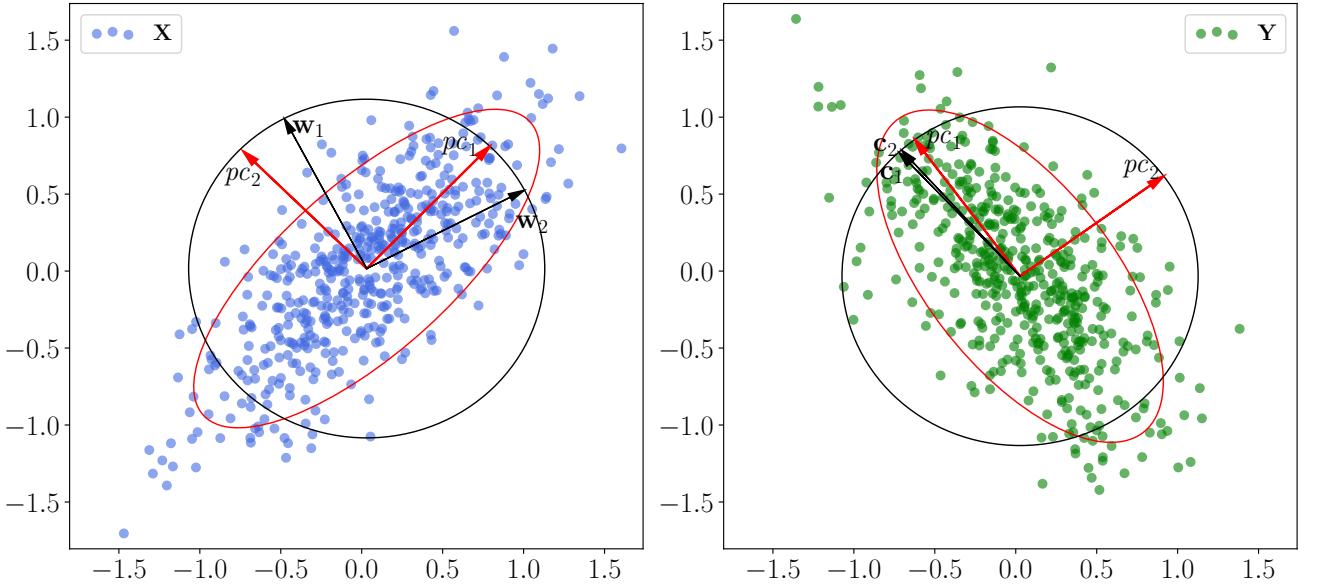


Рис. 1.1: Модельный пример работы алгоритмов PCA и PLS

1.4.3. CCA

Канонический корреляционный анализ (canonical correlation analysis, CCA) широко применяется для поиска взаимосвязи между двумя наборами переменных [27, 28]. Оптимизационная задача CCA похожа на оптимизационную задачу PLS (1.10) с той лишь разницей, что вместо максимизации ковариации CCA максимизирует корреляцию:

$$\max_{\|\mathbf{p}\|_2 = \|\mathbf{q}\|_2 = 1} [\text{corr}(\mathbf{X}\mathbf{p}, \mathbf{Y}\mathbf{q})^2] = \max_{\mathbf{p}, \mathbf{q}} \frac{\mathbf{p}^\top \mathbf{X}^\top \mathbf{Y} \mathbf{q}}{\sqrt{\mathbf{p}^\top \mathbf{X}^\top \mathbf{X} \mathbf{p}} \sqrt{\mathbf{q}^\top \mathbf{Y}^\top \mathbf{Y} \mathbf{q}}}. \quad (1.13)$$

Задача (1.13) может быть также решена с помощью алгоритма 1 с модификацией в шаге (11): $\mathbf{q}_k := \mathbf{Y}_k^\top \mathbf{u}_k / (\mathbf{u}_k^\top \mathbf{u}_k)$. На Рис. 1.2 показан результат работы алгоритма. Основное различие состоит в том, что вектора \mathbf{c}_1 и \mathbf{c}_2 в данном случае становятся ортогональными.

В Таблице 1.1 приведены значения квадратичной ошибки $\mathcal{L}(\Theta, \mathbf{X}, \mathbf{Y})$ для алгоритмов линейной регрессии, PCA и PLS.

Нелинейный ядерный CCA [29, 30, 31, 32] является обобщением базового метода. CCA и ядерный CCA широко используются для задач обучения без учителя [33, 34]. Метод имеет область применения от анализа хемометрических [35]

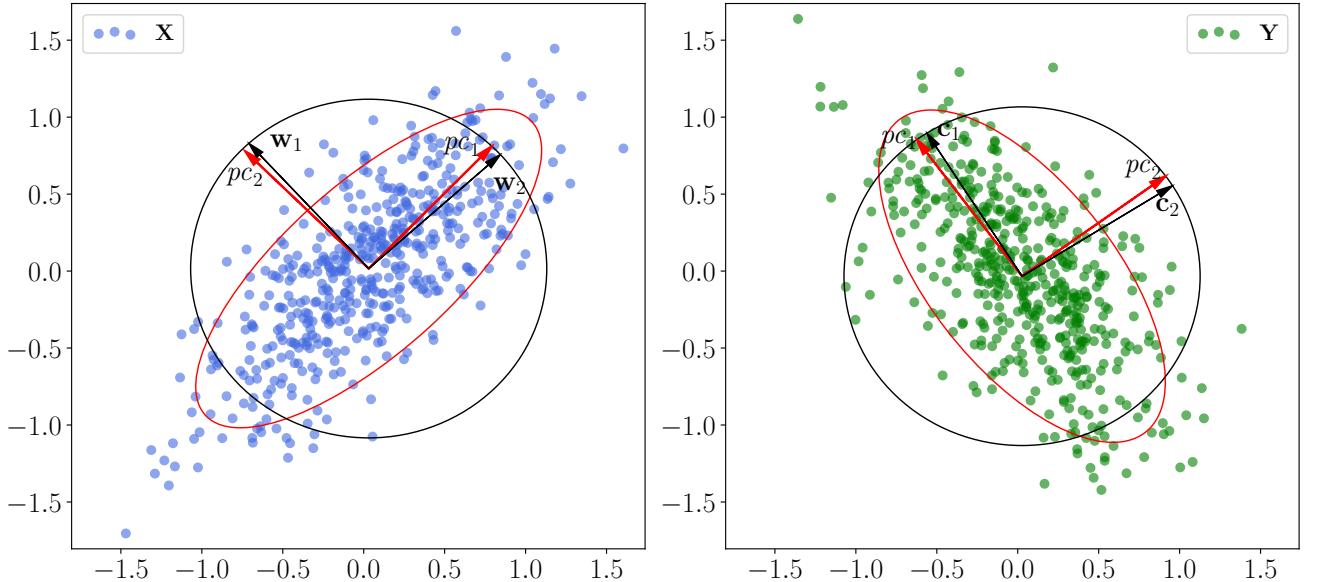


Рис. 1.2: Модельный пример работы алгоритмов PCA и CCA

Линейная регрессия	PCA	PLS	CCA
0.01	0.24	0.13	0.13

Таблица 1.1: Средняя квадратичная ошибка на модельном примере для алгоритмов линейной регрессии, PCA, PLS, CCA

и биологических [36] данных до обработки естественного языка [37, 38], аудиосигналов [39, 40] и компьютерного зрения [41].

В работе [42] был впервые предложено обобщение алгоритма CCA, работающего с нейросетями. Предложенный алгоритм DeepCCA максимизирует корреляцию между представлениями, полученными на выходе нейросети. Принцип работы алгоритма изображен на рисунке 1.3. В статье [43] приведен обширный обзор модификаций нейросетевого CCA для работы с многовидовыми данными. Главным недостатком нейросетевого CCA является вычислительная сложность. В работе [44] предложена релаксация исходного лосса, которая способна масштабироваться под работу с большими глубокими моделями.

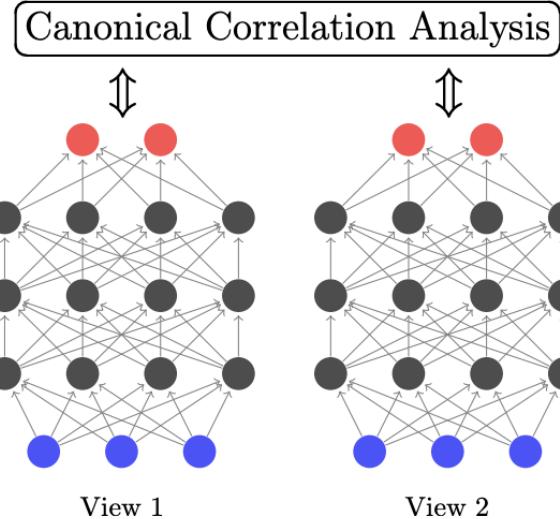


Рис. 1.3: Схематичный пример работы алгоритма DeepCCA из работы [42]

1.4.4. High order PLS/CCA

Исходный объект \mathbf{x} может быть не просто вектором, а быть тензором более высокого порядка. В таком случае для построения модели тензор может быть вытянут в вектор [45]. Но в таком случае модель не будет учитывать имеющиеся зависимости между различными осями исходного тензора. Для учета таких зависимостей используются тензорные версии алгоритма PLS [46, 47, 48].

1.4.5. Многовидовые данные

На практике возникают задачи, где каждый объект имеет несколько представлений. Каждое представление может представлять одну из модальностей. Примерами таких представлений и модальностей могут быть выровненные аудио и видео [49, 50], аудио и артикуляция [51], изображение и текстовая аннотация [32, 52, 53], параллельный корпус текстов [34, 37, 54, 55].

В случае если для каждого объекта имеется более двух представлений, то для построения скрытого пространства для каждого из них применяются два класса подходов. Первый подход состоит в построении скрытого пространства для каждой пары представлений объекта [56, 57]. Второй же подход состоит в

построении общего единого скрытого пространства для всех представлений [58, 59].

Преобразование РСА задается ортогональной матрицей \mathbf{P} . Функция кодирования объектов имеет вид $\varphi_e(\mathbf{x}) = \mathbf{P}^\top \mathbf{x}$. Функция декодирования объектов имеет вид $\varphi_d(\mathbf{t}) = \mathbf{Pt}$.

Функция кодирования и декодирования ответов являются тождественными преобразованиями $\psi_e(\mathbf{y}) = \mathbf{y} = \mathbf{u} = \psi_d(\mathbf{u})$. Преобразование h является линейным.

Метод РСА не согласует объекты \mathbf{x} и ответы \mathbf{y} .

Для РЛС функции кодирования имеют вид:

$$\varphi_e(\mathbf{x}) = \mathbf{W}_x^\top \mathbf{x}, \quad \psi_e(\mathbf{Y}) = \mathbf{W}_y^\top \mathbf{y}, \quad (1.14)$$

где матрицы весов $\mathbf{W}_x \in \mathbb{R}^{m \times p}$, $\mathbf{W}_y \in \mathbb{R}^{k \times p}$. Столбцы матриц весов \mathbf{w}_x^* и \mathbf{w}_y^* находятся путем максимизации функции согласования $g(\mathbf{Xw}_x, \mathbf{Yw}_y) = cov(\mathbf{Xw}_x, \mathbf{Yw}_y)^2$:

$$(\mathbf{w}_x^*, \mathbf{w}_y^*) = \arg \max_{\|\mathbf{w}_y\|_2 = \|\mathbf{w}_y\|_2 = 1} [cov(\mathbf{Xw}_x, \mathbf{Yw}_y)^2] \quad (1.15)$$

где $cov(\mathbf{Xw}_x, \mathbf{Yw}_y)$ – выборочная ковариация между векторами.

Функции декодирования принимают следующий вид:

$$\varphi_d(\mathbf{t}) = \mathbf{Pt}, \quad \psi_d(\mathbf{u}) = \mathbf{Qu}. \quad (1.16)$$

Канонический корреляционный анализ (canonical correlation analysis, CCA) находит два набора базисных векторов $\{\mathbf{w}_{x_i}\}_{i=1}^p$, $\mathbf{w}_x \in \mathbb{R}^m$ и $\{\mathbf{w}_{y_i}\}_{i=1}^p$, $\mathbf{w}_y \in \mathbb{R}^k$, один для \mathbf{X} и другой для \mathbf{Y} , так что коэффициент корреляции между проекциями переменных на эти базисные векторы была максимальной. Функция

согласования для ССА

$$g(\mathbf{X}\mathbf{w}_x, \mathbf{Y}\mathbf{w}_y) = \text{corr}(\mathbf{X}\mathbf{w}_x, \mathbf{Y}\mathbf{w}_y), \quad (1.17)$$

где $\text{corr}(\mathbf{X}\mathbf{w}_x, \mathbf{Y}\mathbf{w}_y)$ – коэффициент корреляции между векторами.

Таким образом, функции кодирования

$$\varphi_e(\mathbf{x}) = \mathbf{W}_x^\top \mathbf{x}, \quad \psi_e(\mathbf{Y}) = \mathbf{W}_y^\top \mathbf{y}, \quad (1.18)$$

где первые столбцы матриц весов находятся, как вектора максимизирующие функцию согласования g . Далее ищутся вектора, максимизирующие g , но с ограничением, что они не коррелируют с первой парой векторов. Процедура продолжается до тех пор, пока количество векторов не станет равным p .

Для постановки задачи декодирования введём предположения о структурах пространств \mathbb{X} и \mathbb{Y} .

Предположение 1. Рассмотрим случай, когда пространства \mathbb{X} и \mathbb{Y} избыточны. Это означает, что объекты \mathbf{x} и \mathbf{y} живут на некоторых многообразиях низкой размерности. В простейшем случае такие многообразия могут являться линейными подпространствами.

Определение 6. Назовём пространство $\mathbb{T} \subset \mathbb{R}^l$ скрытым пространством для пространства $\mathbb{X} \in \mathbb{R}^n$ ($l \leq n$), если существуют функция $\varphi_e : \mathbb{X} \rightarrow \mathbb{T}$ и функция $\varphi_d : \mathbb{T} \rightarrow \mathbb{X}$ такие что

$$\forall \mathbf{x} \in \mathbb{X} \quad \exists \mathbf{t} \in \mathbb{T} : \varphi_d(\varphi_e(\mathbf{x})) = \varphi_d(\mathbf{t}) = \mathbf{x}.$$

Функцию $\varphi_e(\mathbf{x})$ будем называть *функцией кодирования* объекта \mathbf{x} , функцию $\varphi_d(\mathbf{t})$ будем называть *функцией декодирования*.

Аналогично введём определение скрытого пространства $\mathbb{U} \subset \mathbb{R}^s$ для целевого пространства \mathbb{Y} , функции кодирования $\psi_e : \mathbb{Y} \rightarrow \mathbb{U}$ и декодирования $\psi_d : \mathbb{U} \rightarrow \mathbb{Y}$

$$\forall \mathbf{y} \in \mathbb{Y} \quad \exists \mathbf{u} \in \mathbb{U} : \psi_d(\psi_e(\mathbf{y})) = \psi_d(\mathbf{u}) = \mathbf{y}.$$

Определение 7. Будем считать, что пространство $\mathbb{T} \subset \mathbb{R}^l$ задаёт *внутреннюю структуру* пространства $\mathbb{X} \in \mathbb{R}^n$, если пространство \mathbb{T} является скрытым для пространства \mathbb{X} .

Определение 8. Между пространствами \mathbb{X} и \mathbb{Y} существует *согласующее отображение*, если существуют пространства \mathbb{T} и \mathbb{U} , задающие внутренние структуры для пространств \mathbb{X} и \mathbb{Y} соответственно, и существует *функция согласования* $g : \mathbb{T} \rightarrow \mathbb{U}$, такая что

$$\forall \mathbf{u} \in \mathbb{U} \quad \exists \mathbf{t} : \mathbf{u} = g(\mathbf{t}).$$

Предположение 2. Предположим, что в задаче предсказания (1.1) пространства \mathbb{T} и \mathbb{U} задают внутреннюю структуру пространств \mathbb{X} и \mathbb{Y} . Предположим также, что для данных скрытых пространств \mathbb{T} и \mathbb{U} существует функция согласования $g : \mathbb{T} \rightarrow \mathbb{U}$. Тогда выполнено

$$\forall \mathbf{y} \in \mathbb{Y} \quad \exists \mathbf{x} \in \mathbb{X} : \mathbf{y} = \psi_d(\mathbf{u}) = \psi_d(g(\mathbf{t})) = \psi_d(g(\varphi_e(\mathbf{x}))).$$

Тогда общая схема задачи декодирования принимает вид следующей коммутативной диаграммы:

$$\begin{array}{ccc} \mathbb{X} \subset \mathbb{R}^n & \xrightarrow{f} & \mathbb{Y} \subset \mathbb{R}^r \\ \varphi_e \swarrow \varphi_d & & \downarrow \psi_d \\ \mathbb{T} \subset \mathbb{R}^l & \xrightarrow{h} & \mathbb{U} \subset \mathbb{R}^s \\ \end{array} \quad (1.19)$$

Определение 9. Согласно схеме (1.19), определим модель декодирования $f : \mathbb{X} \rightarrow \mathbb{Y}$ как суперпозицию

$$f = \psi_d \circ g \circ \varphi_e. \quad (1.20)$$

Рассмотрим случай квадратичной функции потерь:

$$\mathcal{L}(f, \mathbf{X}, \mathbf{Y}) = \sum_{i=1}^m \|\mathbf{y}_i - f(\mathbf{x}_i)\|^2.$$

Утверждение 4. Пусть функция декодирования ψ_d модели декодирования f (1.20) Липшицева с константой L . Тогда

$$\mathcal{L}(f, \mathbf{X}, \mathbf{Y}) \leq L \cdot \mathcal{L}(g, \mathbf{T}, \mathbf{U}).$$

Доказательство.

$$\begin{aligned} \mathcal{L}(f, \mathbf{X}, \mathbf{Y}) &= \sum_{i=1}^m \|\psi_d(g(\varphi_e(\mathbf{x}_i))) - \mathbf{y}_i\|^2 = \sum_{i=1}^m \|\psi_d(g(\mathbf{t}_i)) - \psi_d(\mathbf{u}_i)\|^2 \\ &\leq L \sum_{i=1}^m \|g(\mathbf{t}_i) - \mathbf{u}_i\|^2 = L \cdot \mathcal{L}(g, \mathbf{T}, \mathbf{U}). \end{aligned}$$

□

Теорема 1. Рассмотрим две модели:

- Первая модель f_1 доставляет минимум ошибке $\mathcal{L}(f, \mathbf{X}, \mathbf{Y})$

$$f_1 = \arg \min_f \mathcal{L}(f, \mathbf{X}, \mathbf{Y}).$$

При этом $\mathbf{Y} = f_1(\mathbf{X}) + \mathbf{E}_1$, где $\mathbf{E}_1 \in \mathbb{R}^{m \times r}$ — матрица ошибок модели f_1 .

- Вторая модель f_2 — модель декодирования (1.20) с Липшецевой функцией декодирования ψ_d с константой L и функцией согласования, удовлетворяющей условию

$$g = \arg \min_g \mathcal{L}(g, \mathbf{T}, \mathbf{U}).$$

При этом $\mathbf{U} = g(\mathbf{T}) + \mathbf{E}_2$, где $\mathbf{E}_2 \in \mathbb{R}^{m \times s}$ — матрица ошибок функции согласования g .

Пусть для константы Липшица L функции декодирования ψ_d выполнено следующее условие

$$L \leq \frac{\|\mathbf{E}_1\|_F^2}{\|\mathbf{E}_2\|_F^2}. \quad (1.21)$$

Тогда функция ошибки $\mathcal{L}(f_2, \mathbf{X}, \mathbf{Y})$ модели f_2 не превосходит функции ошибки $\mathcal{L}(f_1, \mathbf{X}, \mathbf{Y})$ модели f_1 .

Доказательство.

$$\mathcal{L}(f_1, \mathbf{X}, \mathbf{Y}) = \sum_{i=1}^m \|\mathbf{f}_1(\mathbf{x}_i) - \mathbf{y}_i\|^2 = \|\mathbf{E}_1\|_F^2.$$

$$\mathcal{L}(g, \mathbf{T}, \mathbf{U}) = \sum_{i=1}^m \|\mathbf{g}(\mathbf{t}_i) - \mathbf{u}_i\|^2 = \|\mathbf{E}_2\|_F^2.$$

Воспользуемся утверждением 4 и условием (1.21)

$$\mathcal{L}(f_2, \mathbf{X}, \mathbf{Y}) \leq L \cdot \mathcal{L}(g, \mathbf{T}, \mathbf{U}) = L \|\mathbf{E}_2\|_F^2 \leq \|\mathbf{E}_1\|_F^2 = \mathcal{L}(f_1, \mathbf{X}, \mathbf{Y}).$$

□

Определение 10. Функция $g : \mathbb{R}^l \times \mathbb{R}^l \rightarrow \mathbb{R}$, связывающая два низкоразмерных латентных представления, назовём *функцией согласования*.

где $\varphi_e : \mathbb{R}^m \rightarrow \mathbb{R}^l$ – функция кодирования объектов; $\psi_e : \mathbb{R}^r \rightarrow \mathbb{R}^l$ – функция кодирования ответов; $\varphi_d : \mathbb{R}^l \rightarrow \mathbb{R}^n$ – функция декодирования объектов; $\psi_d : \mathbb{R}^l \rightarrow \mathbb{R}^r$ – функция декодирования ответов; $\mathbf{T} = [\varphi_e(\mathbf{x}_1), \dots, \varphi_e(\mathbf{x}_m)]^\top \in \mathbb{R}^{m \times l}$ и $\mathbf{U} = [\psi_e(\mathbf{y}_1), \dots, \psi_e(\mathbf{y}_m)]^\top \in \mathbb{R}^{m \times l}$ – матрицы представлений данных в латентном пространстве низкой размерности; $g : \mathbb{R}^l \times \mathbb{R}^l \rightarrow \mathbb{R}$ – функция согласования.

Оптимальные параметры $\mathbf{W}_{\varphi_e}^*, \mathbf{W}_{\psi_e}^*$ для функций кодирования φ_e и ψ_e находятся из следующей задачи параметрической оптимизации:

$$(\mathbf{W}_{\varphi_e}^*, \mathbf{W}_{\psi_e}^*) = \arg \max_{(\mathbf{W}_{\varphi_e}, \mathbf{W}_{\psi_e})} [g(\varphi_e(\mathbf{X}; \mathbf{W}_{\varphi_e}), \psi_e(\mathbf{Y}; \mathbf{W}_{\psi_e}))]. \quad (1.22)$$

Изменить Так как параметры функции кодирования подбирались из условия максимизации функции согласования (1.22), то после перехода в латентное пространство между \mathbf{T} и \mathbf{U} существует зависимость

$$\mathbf{U} = h(\mathbf{T}) + \boldsymbol{\eta}, \quad (1.23)$$

где $h : \mathbb{R}^{n \times p} \rightarrow \mathbb{R}^{n \times p}$ – функция регрессионной зависимости, $\boldsymbol{\eta}$ – матрица регрессивных ошибок.

Оптимальная h находится минимизацией функции ошибки. Используем квадратичную функцию ошибки потерь \mathcal{L} на \mathbf{T} и \mathbf{U} :

$$\mathcal{L}(h|\mathbf{T}, \mathbf{U}) = \left\| \mathbf{U}_{n \times p} - h(\mathbf{T}_{m \times p}) \right\|_2^2 \rightarrow \min_h. \quad (1.24)$$

Финальная прогностическая модель имеет вид: $\hat{\mathbf{y}} = \psi_d(h(\varphi_e(\mathbf{x})))$, то есть

$$f = \psi_d \circ h \circ \varphi_e \quad (1.25)$$

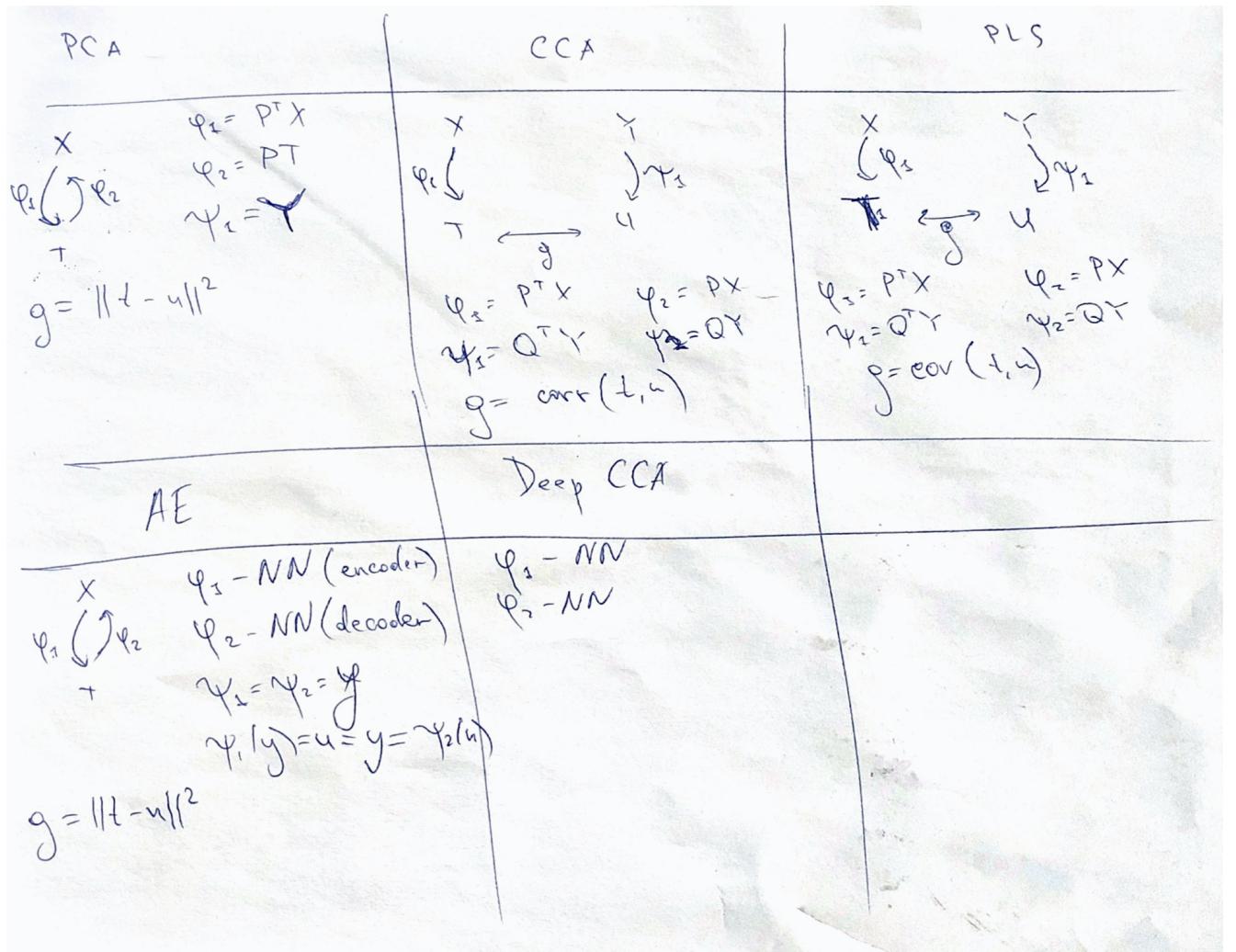


Рис. 1.4: Примеры алгоритмов, работающих по схеме 1.19

1.4.6. Deep CCA

Deep CCA – нелинейной модификация CCA. DCCA преобразует исходные данные с помощью многослойной нейронной сети таким образом, что результи-

рующее представление становится согласованным. Предполагается, что есть d слоев нейронной сети.

Выходом первого слоя для экземпляра \mathbf{x} будет $\mathbf{h}_1 = s(\mathbf{W}_{\mathbf{x}}^1 \mathbf{x} + \mathbf{b}_{\mathbf{x}}^1) \in \mathbb{R}^{c_1}$, где $\mathbf{W}_{\mathbf{x}}^1 \in \mathbb{R}^{c_1 \times m}$ — матрица весов, $\mathbf{b}_{\mathbf{x}}^1 \in \mathbb{R}^{c_1}$ — вектор смещения, $s : \mathbb{R} \rightarrow \mathbb{R}$ — нелинейная функция, которая действует покомпонентно. Далее выход первого слоя используется для вычисления выхода второго слоя $\mathbf{h}_2 = s(\mathbf{W}_{\mathbf{x}}^2 \mathbf{h}_1 + \mathbf{b}_{\mathbf{x}}^2) \in \mathbb{R}^{c_2}$ и так далее до тех пор пока не будет найдено конечное представление $\varphi_e(\mathbf{x}) = s(\mathbf{W}_{\mathbf{x}}^d \mathbf{h}_{d-1} + \mathbf{b}_{\mathbf{x}}^d) \in \mathbb{R}^p$. Аналогично находится представление для \mathbf{y} : $\psi_e(\mathbf{y}) = s(\mathbf{W}_{\mathbf{y}}^d \mathbf{h}_{d-1} + \mathbf{b}_{\mathbf{y}}^d) \in \mathbb{R}^p$.

Обозначим $\theta_{\mathbf{x}}, \theta_{\mathbf{y}}$ — параметры для функций кодирования, то есть матрицы весов и векторы смещений. Оптимальные параметры $\theta_{\mathbf{x}}^*, \theta_{\mathbf{y}}^*$ находятся из задачи оптимизации:

$$(\theta_{\mathbf{x}}^*, \theta_{\mathbf{y}}^*) = \arg \max_{(\theta_{\mathbf{x}}, \theta_{\mathbf{y}})} [g(\varphi_e(\mathbf{X}; \theta_{\mathbf{x}}), \psi_e(\mathbf{Y}; \theta_{\mathbf{y}}))] = \arg \max_{(\theta_{\mathbf{x}}, \theta_{\mathbf{y}})} [\text{corr}(\varphi_e(\mathbf{X}; \theta_{\mathbf{x}}), \psi_e(\mathbf{Y}; \theta_{\mathbf{y}}))]. \quad (1.26)$$

1.5. Вычислительный эксперимент

Временные ряды электроэнергии состоят из почасовых записей (52512 наблюдений). Стока матрицы \mathbf{X} — локальная история сигнала за одну неделю $n = 24 \times 7$. Стока матрицы \mathbf{Y} — локальный прогноз потребления электроэнергии в следующие 24 часа $r = 24$. В этом случае матрицы \mathbf{X} и \mathbf{Y} являются авторегрессионными матрицами.

Вычислительный эксперимент также проводился на данных электрокортикограмм (ECoG) из проекта NeuroTycho [60]. Данные ECoG состоят из 32-канальных сигналов напряжения, снятых с головного мозга. Цель состоит в предсказании по входному сигналу ECoG 3D позиции рук в последующие моменты времени. Исходные сигналы напряжения преобразуются в пространственно-временное представление с помощью вейвлет-преобразования с материнским вейвлетом Морле. Процедура извлечения признаков из исход-

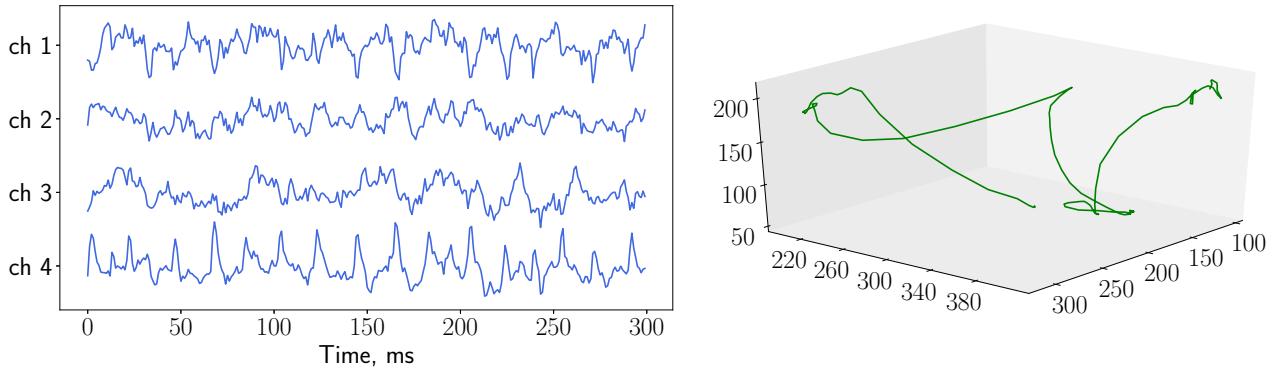


Рис. 1.5: Сигналы мозга (левый график) и 3D координаты руки (правый график)

ных данных подробно описана в [61, 48]. Описание исходного сигнала в каждый момент времени имеет размерность 32 (каналы) \times 27 (частоты) = 864 . Каждый объект представляет собой локальный отрезок времени длительностью $\Delta t = 1s$. Временной шаг между объектами $\delta t = 0.05s$. Матрицы имеют размеры $\mathbf{X} \in \mathbb{R}^{18900 \times 864}$ и $\mathbf{Y} \in \mathbb{R}^{18900 \times 3k}$, где k - число отсчётов времени прогнозирования. Данные разбиты на тренировочную и тестовую части в соотношении $0,67$. Пример исходных сигналов мозга и соответствующей траектории руки показан на рисунке ??.

Введём среднеквадратичную ошибку для некоторых матриц $\mathbf{A} = [a_{ij}]$ и $\mathbf{B} = [b_{ij}]$

$$\text{MSE}(\mathbf{A}, \mathbf{B}) = \sum_{i,j} (a_{ij} - b_{ij})^2.$$

Для оценивания качества аппроксимации вычисляется значение нормированной среднеквадратичной ошибки

$$\text{NMSE}(\mathbf{Y}, \hat{\mathbf{Y}}) = \frac{\text{MSE}(\mathbf{Y}, \hat{\mathbf{Y}})}{\text{MSE}(\mathbf{Y}, \bar{\mathbf{Y}})}, \quad (1.27)$$

где $\hat{\mathbf{Y}}$ — прогноз модели, $\bar{\mathbf{Y}}$ — константный прогноз средним значением по столбцам матрицы.

Данные потребления электроэнергии

Для нахождения оптимальной размерности l латентного пространства все данные потребления электроэнергии были разбиты на обучающую и валидационную части. Обучающая выборка состоит из 700 объектов, валидационная из 370. Зависимость нормированной квадратичной ошибки (1.27) от размерности l латентного пространства представлена на Рис. 1.6. Сначала ошибка резко падает при увеличении размерности скрытого пространства, а затем стабилизируется.

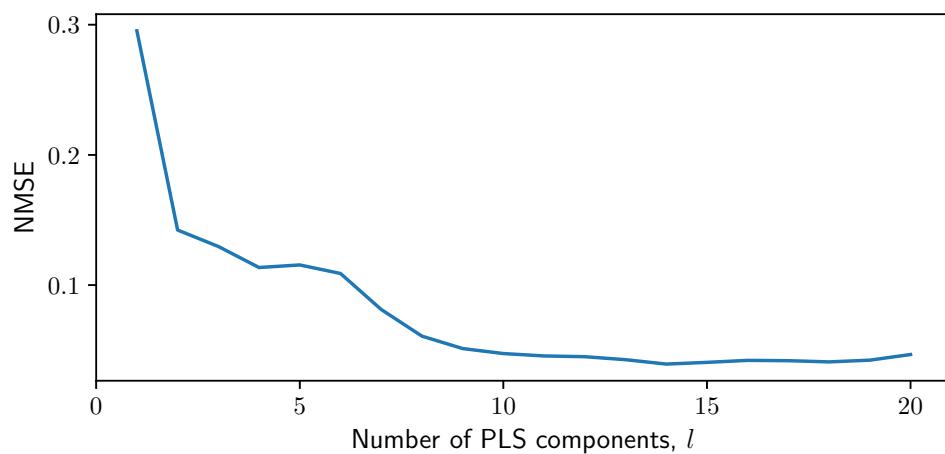


Рис. 1.6: Прогноз потребления электроэнергии алгоритмом PLS при размерности латентного пространства $l=14$

Минимальная ошибка наблюдается при $l = 14$. Построим прогноз потребления электроэнергии при данном l . Результат аппроксимации изображен на Рис. 1.7. Алгоритм PLS восстановил авторегрессионную зависимость и обнаружил дневную сезонность.

Данные электрокортиограммы

На Рис. 1.8 представлена зависимость нормированной квадратичной ошибки (1.27) от размерности латентного пространства. Ошибка аппроксимации меняется незначительно при $l > 5$. Таким образом совместное описание

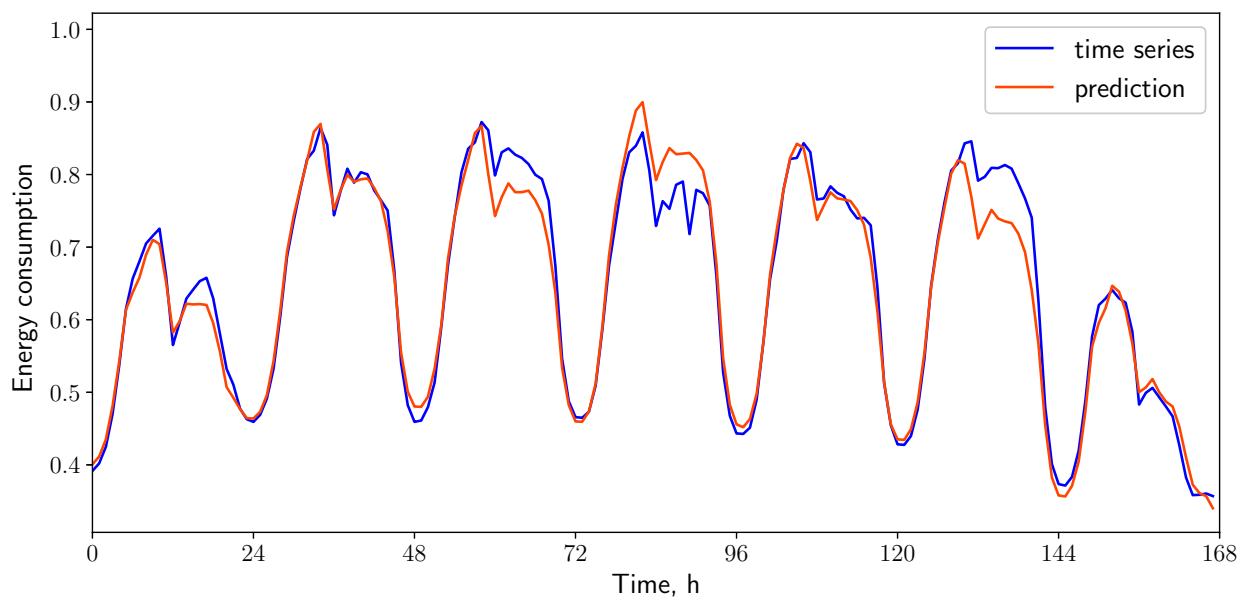


Рис. 1.7: Зависимость ошибки от размерности латентного пространства для данных потребления электроэнергии

пространственно-временного спектрального представления объектов и пространственного положения руки может быть представлено вектором размерности $l \ll n$. Зафиксируем $l = 5$. Пример аппроксимации положения руки изображен на Рис. 1.9. Сплошными линиями изображены истинные координаты руки по всем осям, пунктирными линиями показана аппроксимация методом PLS.

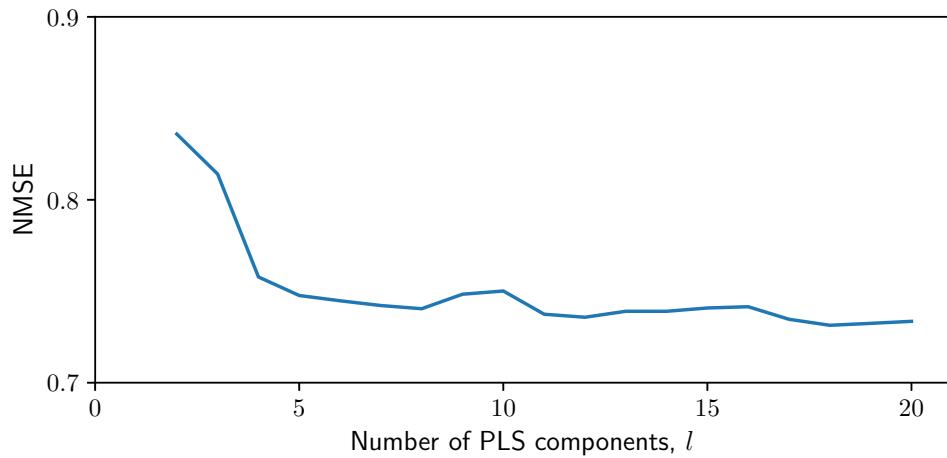


Рис. 1.8: Зависимость ошибки от размерности латентного пространства для данных ECoG

Глава 2

Выбор признаков

Задача выбора признаков заключается в поиске оптимального подмножества признаков \mathcal{A} среди всех возможных $2^n - 1$ вариантов. Существует взаимооднозначное отображение между подмножеством \mathcal{A} и булевым вектором $\mathbf{a} \in \{0, 1\}^n$, компоненты которого указывают, выбран ли признак. Для нахождения оптимального вектора \mathbf{a} введем функцию ошибки выбора признаков $S(\mathbf{a}, \mathbf{X}, \mathbf{Y})$. Проблема выбора признаков принимает вид:

$$\mathbf{a} = \arg \min_{\mathbf{a}' \in \{0,1\}^n} S(\mathbf{a}', \mathbf{X}, \mathbf{Y}). \quad (2.1)$$

Целью выбора признаков является построение функции $S(\mathbf{a}, \mathbf{X}, \mathbf{Y})$. Конкретные примеры данной функции для рассматриваемых алгоритмов выбора признаков приведены ниже и обобщены в таблице 2.1.

Задача (2.1) имеет дискретную область определения $\{0, 1\}^n$. Для решения данной задачи применяется релаксация задачи (2.1) к непрерывной области определения $[0, 1]^n$. Релаксированная задача выбора функции имеет следующий

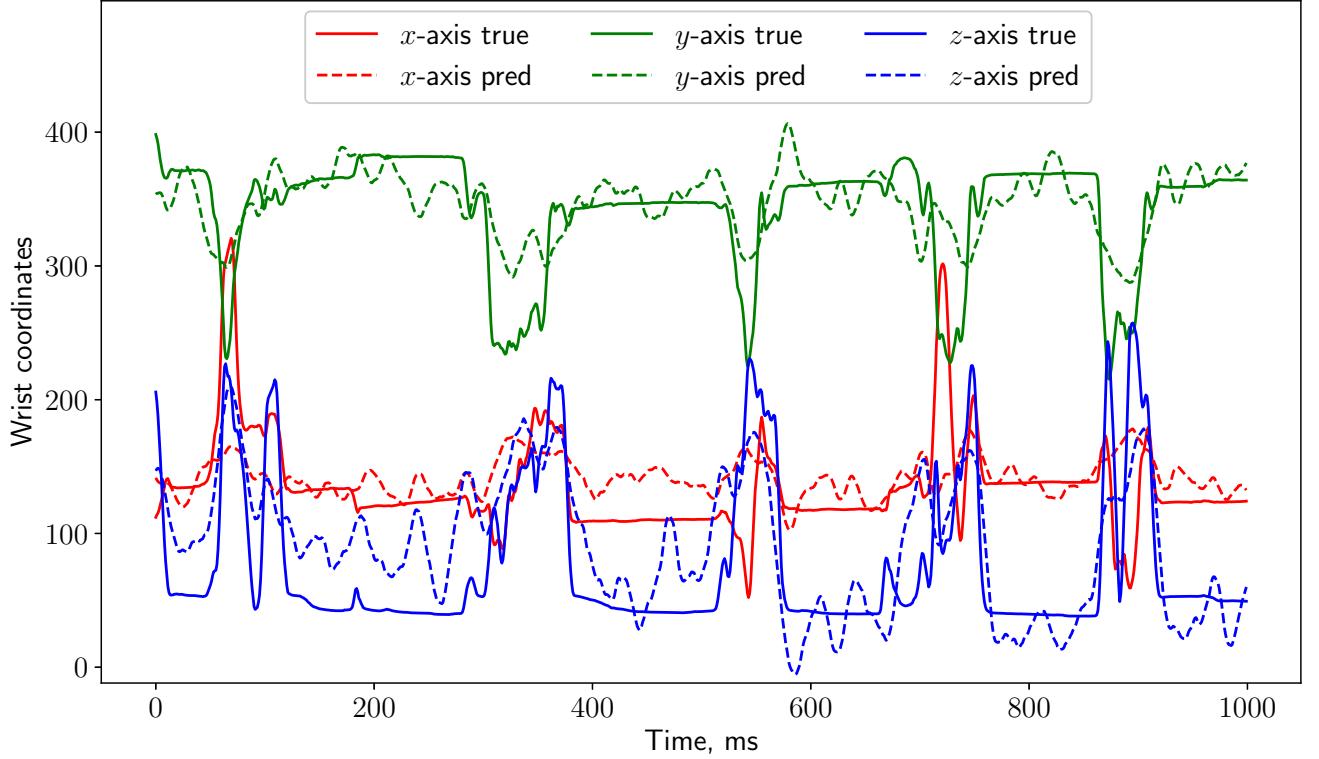


Рис. 1.9: Прогноз движения руки данных ECoG алгоритмом PLS при размерности латентного пространства $l = 5$

вид:

$$\mathbf{z} = \arg \min_{\mathbf{z}' \in [0,1]^n} S(\mathbf{z}', \mathbf{X}, \mathbf{Y}). \quad (2.2)$$

Здесь, компоненты вектора \mathbf{z} – значения нормированных коэффициентов значимости признаков. Сначала решается задача (2.2), для получения вектора значимостей \mathbf{z} . Затем решение (2.1) восстанавливается с помощью отсечения по порогу следующим образом:

$$\mathbf{a} = [a_j]_{j=1}^n, \quad a_j = \begin{cases} 1, & z_j > \tau; \\ 0, & \text{в противном случае.} \end{cases} \quad (2.3)$$

τ – гиперпараметр, который может быть подобран вручную или выбран с помощью кросс-валидации.

Как только решение \mathbf{a} задачи (2.1) получено, задача (??) принимает вид:

$$\mathcal{L}(\Theta_{\mathcal{A}}, \mathbf{X}_{\mathcal{A}}, \mathbf{Y}) = \left\| \mathbf{Y} - \mathbf{X}_{\mathcal{A}} \Theta_{\mathcal{A}}^{\top} \right\|_2^2 \rightarrow \min_{\Theta_{\mathcal{A}}}$$

где индекс \mathcal{A} обозначает подматрицу со столбцами, индексы которых содержатся в \mathcal{A} .

2.1. Выбор признаков с помощью квадратичного программирования

Если между столбцами матрицы плана \mathbf{X} существует линейная зависимость, то решение задачи линейной регрессии

$$\|\boldsymbol{\nu} - \mathbf{X}\boldsymbol{\theta}\|_2^2 \rightarrow \min_{\boldsymbol{\theta} \in \mathbb{R}^n}. \quad (2.4)$$

оказывается неустойчивым. Методы выбора признаков находят подмножество $\mathcal{A} \in \{1, \dots, n\}$ оптимальных столбцов \mathbf{X} .

Алгоритм QPFS выбирает некоррелированные признаки, релевантные целевому вектору $\boldsymbol{\nu}$. Чтобы формализовать этот подход, введем две функции: $\text{Sim}(\mathbf{X})$ и $\text{Rel}(\mathbf{X}, \boldsymbol{\nu})$. $\text{Sim}(\mathbf{X})$ контролирует избыточность между признаками, $\text{Rel}(\mathbf{X}, \boldsymbol{\nu})$ содержит релевантности между каждым признаком и целевым вектором. Мы хотим минимизировать функцию Sim и максимизировать Rel одновременно.

QPFS предлагает явный способ построения функций Sim и Rel . Алгоритм минимизирует следующую функцию ошибки

$$\underbrace{\mathbf{z}^\top \mathbf{Qz}}_{\text{Sim}} - \alpha \cdot \underbrace{\mathbf{b}^\top \mathbf{z}}_{\text{Rel}} \rightarrow \min_{\substack{\mathbf{z} \in \mathbb{R}_+^n \\ \|\mathbf{z}\|_1=1}}. \quad (2.5)$$

Элементы матрицы $\mathbf{Q} \in \mathbb{R}^{n \times n}$ содержат коэффициенты попарного сходства между признаками. Вектор $\mathbf{b} \in \mathbb{R}^n$ выражает сходство между каждым признаком и целевым вектором $\boldsymbol{\nu}$. Нормированный вектор \mathbf{z} отражает значимость каждого признака. Функция ошибки (2.5) штрафует зависимые признаки функцией Sim и штрафует признаки, не релевантные к целевой переменной функцией Rel . Параметр α позволяет контролировать компромисс между Sim и Rel . Авторы оригинальной статьи QPFS [62] предложили способ выбора α , чтобы уравновесить вклад членов $\text{Sim}(\mathbf{X})$ и $\text{Rel}(\mathbf{X}, \boldsymbol{\nu})$

$$\alpha = \frac{\bar{\mathbf{Q}}}{\bar{\mathbf{Q}} + \bar{\mathbf{b}}}, \quad \text{где } \bar{\mathbf{Q}} = \text{mean}(\mathbf{Q}), \quad \bar{\mathbf{b}} = \text{mean}(\mathbf{b}).$$

Чтобы выделить оптимальное подмножество признаков, применяется отсечение по порогу (2.3).

Для измерения сходства используется выборочный коэффициент корреляции Пирсона между парами признаков для функции Sim, и между признаками и целевым вектором для функции Rel:

$$\mathbf{Q} = [|\text{corr}(\boldsymbol{\chi}_i, \boldsymbol{\chi}_j)|]_{i,j=1}^n, \quad \mathbf{b} = [|\text{corr}(\boldsymbol{\chi}_i, \boldsymbol{\nu})|]_{i=1}^n. \quad (2.6)$$

Здесь

$$\text{corr}(\boldsymbol{\chi}, \boldsymbol{\nu}) = \frac{\sum_{i=1}^m (\boldsymbol{\chi}_i - \bar{\boldsymbol{\chi}})(\boldsymbol{\nu}_i - \bar{\boldsymbol{\nu}})}{\sqrt{\sum_{i=1}^m (\boldsymbol{\chi}_i - \bar{\boldsymbol{\chi}})^2 \sum_{i=1}^m (\boldsymbol{\nu}_i - \bar{\boldsymbol{\nu}})^2}}.$$

Другие способы определения \mathbf{Q} и \mathbf{b} рассматриваются в [63]. В работе [63] показано, что алгоритм QPFS превосходит многие существующие алгоритмы выбора функций на различных критериях качества.

Задача (2.5) является выпуклой, если матрица \mathbf{Q} является неотрицательно определенной. В общем случае это не всегда верно. Чтобы удовлетворить этому условию спектр матрицы \mathbf{Q} смещается, и матрица \mathbf{Q} заменяется на $\mathbf{Q} - \lambda_{\min} \mathbf{I}$, где λ_{\min} является минимальным собственным значением \mathbf{Q} .

2.2. Многоиндексный метод выбора признаков

В данном разделе описаны предлагаемые методы выбора признаков для случая нескольких многомерной целевой переменной. В этом случае компоненты целевой переменной могут коррелировать между собой. Предлагаются алгоритмы, учитывающие зависимости как во входном, так и в целевом пространствах.

2.2.1. Агрегация релевантностей целевых переменных

В работе [64], чтобы применить алгоритм QPFS к многомерному случаю ($r > 1$), релевантности признаков агрегируются по всем r компонентам. Член

$\text{Sim}(\mathbf{X})$ остаётся без изменений, матрица \mathbf{Q} определяется как (2.6). Вектор \mathbf{b} агрегируется по всем компонентам целевой переменной и определяется как

$$\mathbf{b} = \left[\sum_{k=1}^r |\text{corr}(\boldsymbol{\chi}_i, \boldsymbol{\nu}_k)| \right]_{i=1}^n.$$

Недостатком такого подхода является отсутствие учёта зависимостей в столбцах матрицы \mathbf{Y} . Рассмотрим следующий пример:

$$\mathbf{X} = [\boldsymbol{\chi}_1, \boldsymbol{\chi}_2, \boldsymbol{\chi}_3], \quad \mathbf{Y} = [\underbrace{\boldsymbol{\nu}_1, \boldsymbol{\nu}_1, \dots, \boldsymbol{\nu}_1}_{r-1}, \boldsymbol{\nu}_2].$$

Пусть матрица \mathbf{X} содержит 3 столбца, матрица $\mathbf{Y} - r$ столбцов, где первые $r - 1$ компонент целевой переменной идентичны. Попарные сходства признаков задаются матрицей \mathbf{Q} . Матрица \mathbf{B} содержит попарные сходства признаков и целевых столбцов. Вектор \mathbf{b} получен суммированием матрицы \mathbf{B} по столбцами

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0.8 \\ 0 & 0.8 & 1 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0.4 & \dots & 0.4 & 0 \\ 0.5 & \dots & 0.5 & 0.8 \\ 0.8 & \dots & 0.8 & 0.1 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} (r-1) \cdot 0.4 + 0 \\ (r-1) \cdot 0.5 + 0.8 \\ (r-1) \cdot 0.8 + 0.1 \end{bmatrix}. \quad (2.7)$$

Пусть необходимо выбрать только 2 признака. В данном случае оптимальным подмножеством признаков является $[\boldsymbol{\chi}_1, \boldsymbol{\chi}_2]$. Признак $\boldsymbol{\chi}_2$ предсказывает второй целевой столбец $\boldsymbol{\nu}_2$, комбинация признаков $\boldsymbol{\chi}_1, \boldsymbol{\chi}_2$ прогнозирует первый целевой столбец $\boldsymbol{\nu}_1$. Алгоритм QPFS для $r = 2$ дает решение $\mathbf{z} = [0.37, 0.61, 0.02]$. Это совпадает с описанным решением. Однако, если добавить коллинеарные столбцы в матрицу \mathbf{Y} и увеличить r до 5, то решением QPFS будет $\mathbf{z} = [0.40, 0.17, 0.43]$. Здесь потерян признак $\boldsymbol{\chi}_2$ и выбран избыточный признак $\boldsymbol{\chi}_3$. В следующих подразделах предлагаются обобщения алгоритма QPFS, которые позволяют бороться с проблемой данного примера.

2.2.2. Симметричный учёт значимости признаков и целевых переменных

Чтобы учесть зависимости в столбцах матрицы \mathbf{Y} , обобщим функцию QPFS (2.5) для многомерного случая ($r > 1$). Добавим член $\text{Sim}(\mathbf{Y})$ и изменим член $\text{Rel}(\mathbf{X}, \mathbf{Y})$ следующим образом:

$$\alpha_1 \cdot \underbrace{\mathbf{z}_x^\top \mathbf{Q}_x \mathbf{z}_x}_{\text{Sim}(\mathbf{X})} - \alpha_2 \cdot \underbrace{\mathbf{z}_x^\top \mathbf{B} \mathbf{z}_y}_{\text{Rel}(\mathbf{X}, \mathbf{Y})} + \alpha_3 \cdot \underbrace{\mathbf{z}_y^\top \mathbf{Q}_y \mathbf{z}_y}_{\text{Sim}(\mathbf{Y})} \rightarrow \min_{\substack{\mathbf{z}_x \geq \mathbf{0}_n, \mathbf{1}_n^\top \mathbf{z}_x = 1 \\ \mathbf{z}_y \geq \mathbf{0}_r, \mathbf{1}_r^\top \mathbf{z}_y = 1}} . \quad (2.8)$$

Определим элементы матриц $\mathbf{Q}_x \in \mathbb{R}^{n \times n}$, $\mathbf{Q}_y \in \mathbb{R}^{r \times r}$ и $\mathbf{B} \in \mathbb{R}^{n \times r}$ следующим образом:

$$\mathbf{Q}_x = [|\text{corr}(\boldsymbol{\chi}_i, \boldsymbol{\chi}_j)|]_{i,j=1}^n, \quad \mathbf{Q}_y = [|\text{corr}(\boldsymbol{\nu}_i, \boldsymbol{\nu}_j)|]_{i,j=1}^r, \quad \mathbf{B} = [|\text{corr}(\boldsymbol{\chi}_i, \boldsymbol{\nu}_j)|]_{i=1,\dots,n, j=1,\dots,r}.$$

Вектор \mathbf{z}_x содержит коэффициенты значимости признаков, \mathbf{z}_y – коэффициенты значимости целевых столбцов. Коррелированные целевые столбцы штрафуются членом $\text{Sim}(\mathbf{Y})$ и получают более низкие значения значимости.

Коэффициенты α_1 , α_2 , и α_3 контролируют влияние каждого члена на функцию (2.8) и удовлетворяют следующим условиям:

$$\alpha_1 + \alpha_2 + \alpha_3 = 1, \quad \alpha_i \geq 0, \quad i = 1, 2, 3.$$

Утверждение 5. Баланс между $\text{Sim}(\mathbf{X})$, $\text{Rel}(\mathbf{X}, \mathbf{Y})$ и $\text{Sim}(\mathbf{Y})$ в задаче (2.8) достигается при:

$$\alpha_1 \propto \overline{\mathbf{Q}_y \mathbf{B}}; \quad \alpha_2 \propto \overline{\mathbf{Q}_x \mathbf{Q}_y}; \quad \alpha_3 \propto \overline{\mathbf{Q}_x \mathbf{B}}. \quad (2.9)$$

Доказательство. Значения α_1 , α_2 , и α_3 получаются путем решения следующих уравнений:

$$\alpha_1 + \alpha_2 + \alpha_3 = 1;$$

$$\alpha_1 \overline{\mathbf{Q}_x} = \alpha_2 \overline{\mathbf{B}} = \alpha_3 \overline{\mathbf{Q}_y}.$$

Здесь $\overline{\mathbf{Q}_x}$, $\overline{\mathbf{B}}$ и $\overline{\mathbf{Q}_y}$ соответствующих матриц \mathbf{Q}_x , \mathbf{B} и \mathbf{Q}_y – средние значения членов $\text{Sim}(\mathbf{X})$, $\text{Rel}(\mathbf{X}, \mathbf{Y})$ и $\text{Sim}(\mathbf{Y})$. \square

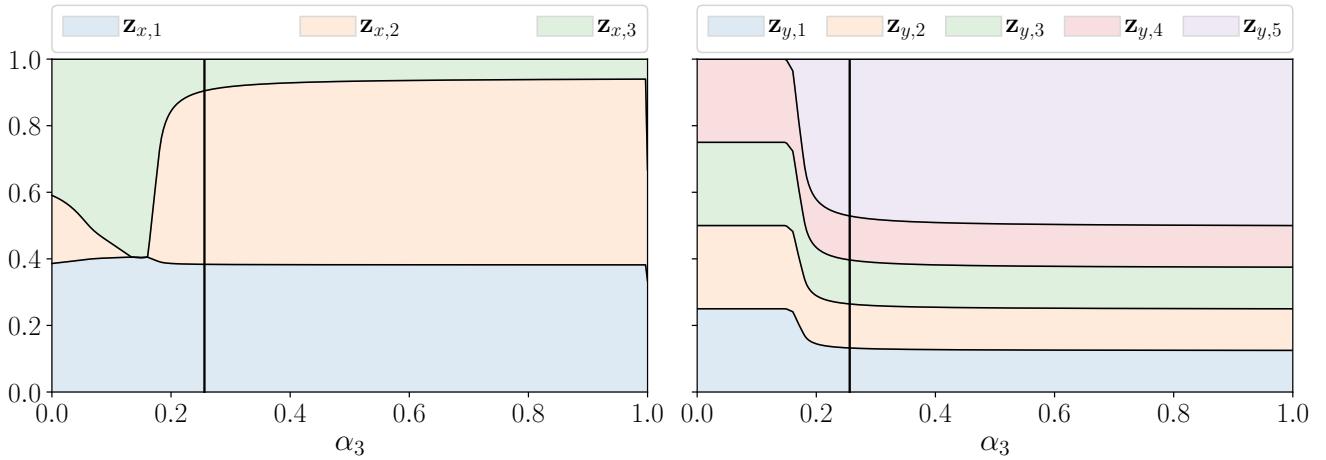


Рис. 2.1: Значимости признаков \mathbf{z}_x и целевых векторов \mathbf{z}_y в зависимости от α_3 для рассмотренного примера

Для изучения зависимости $\text{Sim}(\mathbf{Y})$ на функцию (2.8), зафиксируем соотношение между α_1 и α_2 :

$$\alpha_1 = \frac{(1 - \alpha_3)\bar{\mathbf{B}}}{\bar{\mathbf{Q}}_x + \bar{\mathbf{B}}}; \quad \alpha_2 = \frac{(1 - \alpha_3)\bar{\mathbf{Q}}_x}{\bar{\mathbf{Q}}_x + \bar{\mathbf{B}}}; \quad \alpha_3 \in [0, 1]. \quad (2.10)$$

Применим предложенный алгоритм к приведенному примеру (2.7). Матрица \mathbf{Q} соответствует матрице \mathbf{Q}_x . Определим матрицы \mathbf{Q}_y как $\text{corr}(\boldsymbol{\nu}_1, \boldsymbol{\nu}_2) = 0.2$, а все остальные элементы зададим 1. Рисунок 2.1 показывает значение векторов значимостей признаков \mathbf{z}_x и целевых векторов \mathbf{z}_y в зависимости от значения коэффициента α_3 . Если α_3 мало, значимости всех целевых векторов не различимы и значимость признака χ_3 выше значимости признака χ_2 . При увеличении α_3 до 0.2, коэффициент значимости $\mathbf{z}_{y,5}$ целевого вектора $\boldsymbol{\nu}_5$ увеличивается наряду со значимостью признака χ_2 .

2.2.3. Минимаксная постановка задачи выбора признаков

Функция (2.8) является симметричной по отношению к \mathbf{z}_x и \mathbf{z}_y . Она штрафует признаки, которые коррелированы и не имеют отношения к целевым векторам. Кроме того, она штрафует целевые векторы, которые коррелированы между собой и недостаточно коррелируют с признаками. Это приводит к ма-

лым значениям значимостей для целевых векторов, которые слабо коррелируют с признаками, и большим значениям для целевых векторов, которые сильно коррелируют с признаками. Этот результат противоречит интуиции. Цель — предсказать все целевые вектора, особенно те, которые слабо коррелируют с признаками. Сформулируем две взаимосвязанные задачи:

$$\alpha_1 \cdot \underbrace{\mathbf{z}_x^\top \mathbf{Q}_x \mathbf{z}_x}_{\text{Sim}(\mathbf{X})} - \alpha_2 \cdot \underbrace{\mathbf{z}_x^\top \mathbf{B} \mathbf{z}_y}_{\text{Rel}(\mathbf{X}, \mathbf{Y})} \rightarrow \min_{\substack{\mathbf{z}_x \geq \mathbf{0}_n, \\ \mathbf{1}_n^\top \mathbf{z}_x = 1}} ; \quad (2.11)$$

$$\alpha_3 \cdot \underbrace{\mathbf{z}_y^\top \mathbf{Q}_y \mathbf{z}_y}_{\text{Sim}(\mathbf{Y})} + \alpha_2 \cdot \underbrace{\mathbf{z}_x^\top \mathbf{B} \mathbf{z}_y}_{\text{Rel}(\mathbf{X}, \mathbf{Y})} \rightarrow \min_{\substack{\mathbf{z}_y \geq \mathbf{0}_r, \\ \mathbf{1}_r^\top \mathbf{z}_y = 1}} . \quad (2.12)$$

Разница между (2.11) и (2.12) заключается в знаке перед членом Rel. В пространстве входных объектов нерелевантные признаки должны иметь меньшие значения значимости. В то же время целевые вектора, не релевантные признакам, должны иметь большую значимость. Задачи (2.11) и (2.12) объединяются в совместную минимакс или максмин постановку

$$\min_{\substack{\mathbf{z}_x \geq \mathbf{0}_n \\ \mathbf{1}_n^\top \mathbf{z}_x = 1}} \max_{\substack{\mathbf{z}_y \geq \mathbf{0}_r \\ \mathbf{1}_r^\top \mathbf{z}_y = 1}} f(\mathbf{z}_x, \mathbf{z}_y), \quad \left(\text{или } \max_{\substack{\mathbf{z}_y \geq \mathbf{0}_r \\ \mathbf{1}_r^\top \mathbf{z}_y = 1}} \min_{\substack{\mathbf{z}_x \geq \mathbf{0}_n \\ \mathbf{1}_n^\top \mathbf{z}_x = 1}} f(\mathbf{z}_x, \mathbf{z}_y) \right), \quad (2.13)$$

где

$$f(\mathbf{z}_x, \mathbf{z}_y) = \alpha_1 \cdot \underbrace{\mathbf{z}_x^\top \mathbf{Q}_x \mathbf{z}_x}_{\text{Sim}(\mathbf{X})} - \alpha_2 \cdot \underbrace{\mathbf{z}_x^\top \mathbf{B} \mathbf{z}_y}_{\text{Rel}(\mathbf{X}, \mathbf{Y})} - \alpha_3 \cdot \underbrace{\mathbf{z}_y^\top \mathbf{Q}_y \mathbf{z}_y}_{\text{Sim}(\mathbf{Y})} .$$

Теорема 2. Для положительно определенной матрицы \mathbf{Q}_x и \mathbf{Q}_y , максмин и минимакс задачи (2.13) имеют одинаковое оптимальное значение.

Доказательство. Введём обозначения

$$\mathbb{C}^n = \{\mathbf{z} : \mathbf{z} \geq \mathbf{0}_n, \mathbf{1}_n^\top \mathbf{z} = 1\}, \quad \mathbb{C}^r = \{\mathbf{z} : \mathbf{z} \geq \mathbf{0}_r, \mathbf{1}_r^\top \mathbf{z} = 1\}.$$

Множества \mathbb{C}^n и \mathbb{C}^r — компактные и выпуклые. Функция $f : \mathbb{C}^n \times \mathbb{C}^r \rightarrow \mathbb{R}$ является непрерывной. Если \mathbf{Q}_x и \mathbf{Q}_y положительно определены, функция f выпукло-вогнутая. Т. е., $f(\cdot, \mathbf{z}_y) : \mathbb{C}^n \rightarrow \mathbb{R}$ выпуклая при фиксированном \mathbf{z}_y ,

а $f(\mathbf{z}_x, \cdot) : \mathbb{C}^r \rightarrow \mathbb{R}$ вогнута при фиксированном \mathbf{z}_x . В этом случае по теореме Неймана о минимаксе

$$\min_{\mathbf{z}_x \in \mathbb{C}^n} \max_{\mathbf{z}_y \in \mathbb{C}^r} f(\mathbf{z}_x, \mathbf{z}_y) = \max_{\mathbf{z}_y \in \mathbb{C}^r} \min_{\mathbf{z}_x \in \mathbb{C}^n} f(\mathbf{z}_x, \mathbf{z}_y).$$

□

Для решения минимакс задачи (2.13), зафиксируем некоторый $\mathbf{z}_x \in \mathbb{C}^n$. Для фиксированного вектора \mathbf{z}_x решаем задачу

$$\max_{\mathbf{z}_y \in \mathbb{C}^r} f(\mathbf{z}_x, \mathbf{z}_y) = \max_{\substack{\mathbf{z}_y \geq \mathbf{0}_r \\ \mathbf{1}_r^\top \mathbf{z}_y = 1}} [\alpha_1 \cdot \mathbf{z}_x^\top \mathbf{Q}_x \mathbf{z}_x - \alpha_2 \cdot \mathbf{z}_x^\top \mathbf{B} \mathbf{z}_y - \alpha_3 \cdot \mathbf{z}_y^\top \mathbf{Q}_y \mathbf{z}_y]. \quad (2.14)$$

Лагранжиан для данной задачи:

$$L(\mathbf{z}_x, \mathbf{z}_y, \lambda, \boldsymbol{\mu}) = \alpha_1 \cdot \mathbf{z}_x^\top \mathbf{Q}_x \mathbf{z}_x - \alpha_2 \cdot \mathbf{z}_x^\top \mathbf{B} \mathbf{z}_y - \alpha_3 \cdot \mathbf{z}_y^\top \mathbf{Q}_y \mathbf{z}_y + \lambda \cdot (\mathbf{1}_r^\top \mathbf{z}_y - 1) + \boldsymbol{\mu}^\top \mathbf{z}_y.$$

Здесь вектор множителей Лагранжа $\boldsymbol{\mu}$, который соответствует ограничениям на неравенства $\mathbf{z}_y \geq \mathbf{0}_r$, является неотрицательным. Двойственной задачей является

$$\min_{\lambda, \boldsymbol{\mu} \geq \mathbf{0}_r} g(\mathbf{z}_x, \lambda, \boldsymbol{\mu}) = \min_{\lambda, \boldsymbol{\mu} \geq \mathbf{0}_r} \left[\max_{\mathbf{z}_y \in \mathbb{R}^r} L(\mathbf{z}_x, \mathbf{z}_y, \lambda, \boldsymbol{\mu}) \right]. \quad (2.15)$$

Для задачи квадратичного программирования (2.14) с положительно определенными матрицами \mathbf{Q}_x и \mathbf{Q}_y выполняются условия сильной двойственности. Таким образом, оптимальное значение (2.14) равно оптимальному значению (2.15). Это позволяет перейти от решения задачи (2.13) к решению задачи

$$\min_{\mathbf{z}_x \in \mathbb{C}^n, \lambda, \boldsymbol{\mu} \geq \mathbf{0}_r} g(\mathbf{z}_y, \lambda, \boldsymbol{\mu}). \quad (2.16)$$

Полагая градиент $\nabla_{\mathbf{z}_y} L(\mathbf{z}_x, \mathbf{z}_y, \lambda, \boldsymbol{\mu})$ равным нулю, получим оптимальное значение \mathbf{z}_y :

$$\mathbf{z}_y = \frac{1}{2\alpha_3} \mathbf{Q}_y^{-1} \left(-\alpha_2 \cdot \mathbf{B}^\top \mathbf{z}_x + \lambda \cdot \mathbf{1}_r + \boldsymbol{\mu} \right). \quad (2.17)$$

Двойственная функция принимает вид

$$\begin{aligned} g(\mathbf{z}_x, \lambda, \boldsymbol{\mu}) = \max_{\mathbf{z}_y \in \mathbb{R}^r} L(\mathbf{z}_x, \mathbf{z}_y, \lambda, \boldsymbol{\mu}) &= \mathbf{z}_x^\top \left(-\frac{\alpha_2^2}{4\alpha_3} \cdot \mathbf{B} \mathbf{Q}_y^{-1} \mathbf{B}^\top - \alpha_1 \cdot \mathbf{Q}_x \right) \mathbf{z}_x \\ &\quad - \frac{1}{4\alpha_3} \lambda^2 \cdot \mathbf{1}_r^\top \mathbf{Q}_y^{-1} \mathbf{1}_r - \frac{1}{4\alpha_3} \cdot \boldsymbol{\mu}^\top \mathbf{Q}_y^{-1} \boldsymbol{\mu} + \frac{\alpha_2}{2\alpha_3} \lambda \cdot \mathbf{1}_r^\top \mathbf{Q}_y^{-1} \mathbf{B}^\top \mathbf{z}_x \\ &\quad - \frac{1}{2\alpha_3} \lambda \cdot \mathbf{1}_r^\top \mathbf{Q}_y^{-1} \boldsymbol{\mu} + \frac{\alpha_2}{2\alpha_3} \cdot \boldsymbol{\mu}^\top \mathbf{Q}_y^{-1} \mathbf{B}^\top \mathbf{z}_x + \lambda. \end{aligned} \quad (2.18)$$

Тем самым задача (2.16) является квадратичной задачей с $n + r + 1$ переменными.

2.2.4. Несимметричный учёт значимостей признаков и целевых переменных

Естественным способом преодоления проблемы алгоритма SymImp является добавление штрафа для целевых векторов, которые коррелируют с признаками.

Добавим линейный член $\mathbf{b}^\top \mathbf{z}_y$ в член $\text{Rel}(\mathbf{X}, \mathbf{Y})$ следующим образом:

$$\alpha_1 \cdot \underbrace{\mathbf{z}_x^\top \mathbf{Q}_x \mathbf{z}_x}_{\text{Sim}(\mathbf{X})} - \alpha_2 \cdot \underbrace{\left(\mathbf{z}_x^\top \mathbf{B} \mathbf{z}_y - \mathbf{b}^\top \mathbf{z}_y \right)}_{\text{Rel}(\mathbf{X}, \mathbf{Y})} + \alpha_3 \cdot \underbrace{\mathbf{z}_y^\top \mathbf{Q}_y \mathbf{z}_y}_{\text{Sim}(\mathbf{Y})} \rightarrow \min_{\substack{\mathbf{z}_x \geq \mathbf{0}_n, \mathbf{1}_n^\top \mathbf{z}_x = 1 \\ \mathbf{z}_y \geq \mathbf{0}_r, \mathbf{1}_r^\top \mathbf{z}_y = 1}}. \quad (2.19)$$

Утверждение 6. Пусть вектор \mathbf{b} равен

$$b_j = \max_{i=1, \dots, n} [\mathbf{B}]_{i,j}.$$

Тогда значение коэффициентов значимостей вектора \mathbf{z}_y будут неотрицательными в $\text{Rel}(\mathbf{X}, \mathbf{Y})$ для задачи (2.19).

Доказательство. Утверждение следует из факта

$$\sum_{i=1}^n z_i b_{ij} \leq \left(\sum_{i=1}^n z_i \right) \max_{i=1, \dots, n} b_{ij} = \max_{i=1, \dots, n} b_{ij},$$

где $z_i \geq 0$ и $\sum_{i=1}^n z_i = 1$. □

Следовательно, функция (2.19) штрафует в меньшей мере признаки, которые имеют отношение к целевым векторам, и целевые векторы, которые недостаточно коррелированы с признаками.

Утверждение 7. Баланс между членами $\text{Sim}(\mathbf{X})$, $\text{Rel}(\mathbf{X}, \mathbf{Y})$ и $\text{Rel}(\mathbf{X}, \mathbf{Y})$ для задачи (2.19) достигается при следующих коэффициентах:

$$\alpha_1 \propto \overline{\mathbf{Q}}_y (\overline{\mathbf{b}} - \overline{\mathbf{B}}); \quad \alpha_2 \propto \overline{\mathbf{Q}}_x \overline{\mathbf{Q}}_y; \quad \alpha_3 \propto \overline{\mathbf{Q}}_x \overline{\mathbf{B}}.$$

Доказательство. Необходимые значения α_1 , α_2 , и α_3 являются решением следующей системы уравнений:

$$\alpha_1 + \alpha_2 + \alpha_3 = 1; \quad (2.20)$$

$$\alpha_1 \overline{\mathbf{Q}}_x = \alpha_2 \overline{\mathbf{B}}; \quad (2.21)$$

$$\alpha_2 (\overline{\mathbf{b}} - \overline{\mathbf{B}}) = \alpha_3 \overline{\mathbf{Q}}_y. \quad (2.22)$$

Здесь, в (2.21) уравновешены $\text{Sim}(\mathbf{X})$ с первым слагаемым $\text{Rel}(\mathbf{X}, \mathbf{Y})$, а в (2.22) уравновешены $\text{Sim}(\mathbf{Y})$ с $\text{Rel}(\mathbf{X}, \mathbf{Y})$. \square

Утверждение 8. Для случая $r = 1$, предложенные функции (2.8), (2.13) и (2.19) совпадают с оригинальным алгоритмом QPFS (2.5).

Доказательство. Если r равно 1, то $\mathbf{Q}_y = q_y$ - скаляр, $\mathbf{z}_y = 1$ и $\mathbf{B} = \mathbf{b}$. Задачи (2.8), (2.13) и (2.19) принимают вид

$$\alpha_1 \cdot \mathbf{z}_x^\top \mathbf{Q}_x \mathbf{z}_x - \alpha_2 \cdot \mathbf{z}_x^\top \mathbf{b} \rightarrow \min_{\mathbf{z}_x \geq \mathbf{0}_n, \mathbf{1}_n^\top \mathbf{z}_x = 1}.$$

При $\alpha = \frac{\alpha_2}{\alpha_1 + \alpha_2}$ последняя задача принимает вид (2.5). \square

Таблица 2.1 демонстрирует основные идеи и функции ошибок для каждого алгоритма. RelAgg является базовой стратегией и не учитывает корреляции в целевом пространстве. SymImp штрафует попарные корреляции между целевыми векторами. MinMax более чувствителен к целевым векторам, которые трудно предсказать. Стратегия AsymImp добавляет линейный член к функции SymImp, чтобы сделать вклад признаков и целевых векторов асимметричным.

Таблица 2.1: Обзор предлагаемых обобщений многомерного QPFS алгоритма

Алгоритм	Идея	Функция ошибки $S(\mathbf{z} \mathbf{X}, \mathbf{Y})$
RelAgg	$\min [\text{Sim}(\mathbf{X}) - \text{Rel}(\mathbf{X}, \mathbf{Y})]$	$\min_{\mathbf{z}_x} [(1 - \alpha) \cdot \mathbf{z}_x^\top \mathbf{Q}_x \mathbf{z}_x - \alpha \cdot \mathbf{z}_x^\top \mathbf{B} \mathbf{1}_r]$
SymImp	$\min [\text{Sim}(\mathbf{X}) - \text{Rel}(\mathbf{X}, \mathbf{Y}) + \text{Sim}(\mathbf{Y})]$	$\min_{\mathbf{z}_x, \mathbf{z}_y} [\alpha_1 \cdot \mathbf{z}_x^\top \mathbf{Q}_x \mathbf{z}_x - \alpha_2 \cdot \mathbf{z}_x^\top \mathbf{B} \mathbf{z}_y + \alpha_3 \cdot \mathbf{z}_y^\top \mathbf{Q}_y \mathbf{z}_y]$
MinMax	$\min [\text{Sim}(\mathbf{X}) - \text{Rel}(\mathbf{X}, \mathbf{Y})]$ $\max [\text{Rel}(\mathbf{X}, \mathbf{Y}) + \text{Sim}(\mathbf{Y})]$	$\min_{\mathbf{z}_x} \max_{\mathbf{z}_y} [\alpha_1 \cdot \mathbf{z}_x^\top \mathbf{Q}_x \mathbf{z}_x - \alpha_2 \cdot \mathbf{z}_x^\top \mathbf{B} \mathbf{z}_y - \alpha_3 \cdot \mathbf{z}_y^\top \mathbf{Q}_y \mathbf{z}_y]$
AsymImp	$\min [\text{Sim}(\mathbf{X}) - \text{Rel}(\mathbf{X}, \mathbf{Y})]$ $\max [\text{Rel}(\mathbf{X}, \mathbf{Y}) + \text{Sim}(\mathbf{Y})]$	$\min_{\mathbf{z}_x, \mathbf{z}_y} [\alpha_1 \cdot \mathbf{z}_x^\top \mathbf{Q}_x \mathbf{z}_x - \alpha_2 \cdot (\mathbf{z}_x^\top \mathbf{B} \mathbf{z}_y - \mathbf{b}^\top \mathbf{z}_y) + \alpha_3 \cdot \mathbf{z}_y^\top \mathbf{Q}_y \mathbf{z}_y]$

2.3. Вычислительный эксперимент

Для оценки предложенных алгоритмов выбора признаков, введём критерии оценки качества выбранного количества признаков. Определим коэффициент мультикорреляции как среднее значение коэффициента множественной корреляции следующим образом:

$$R^2 = \frac{1}{r} \text{tr} \left(\mathbf{C}^\top \mathbf{R}^{-1} \mathbf{C} \right); \quad \text{where } \mathbf{C} = [\text{corr}(\boldsymbol{\chi}_i, \boldsymbol{\nu}_j)]_{i=1, \dots, n, j=1, \dots, r}, \mathbf{R} = [\text{corr}(\boldsymbol{\chi}_i, \boldsymbol{\chi}_j)]_{i,j=1}^n.$$

Этот коэффициент принимает значение между 0 и 1. Большее значение R^2 соответствует лучшему подмножеству признаков.

Нормированный среднеквадратичная ошибка (sRMSE) отображает качество прогнозирования модели. Оценка sRMSE считается на тренировочной и тестовой выборке.

$$\text{sRMSE}(\mathbf{Y}, \widehat{\mathbf{Y}}_{\mathbf{a}}) = \sqrt{\frac{\text{MSE}(\mathbf{Y}, \widehat{\mathbf{Y}}_{\mathbf{a}})}{\text{MSE}(\mathbf{Y}, \overline{\mathbf{Y}})}} = \frac{\|\mathbf{Y} - \widehat{\mathbf{Y}}_{\mathbf{a}}\|_2}{\|\mathbf{Y} - \overline{\mathbf{Y}}\|_2}.$$

Здесь $\widehat{\mathbf{Y}}_{\mathbf{a}} = \mathbf{X}_{\mathbf{a}} \boldsymbol{\Theta}_{\mathbf{a}}^\top$ – предсказание модель, $\overline{\mathbf{Y}}$ – предсказание константной модели, полученное усреднением целевой переменной по всем объектам. Данный

показатель на тестовой выборке необходимо минимизировать.

Байесовский информационный критерий (BIC) – компромисс между качеством предсказания и размером выбранного подмножества признаков $\|\mathbf{a}\|_0 = \#\{j : a_j \neq 0\} = \sum_{j=1}^n a_j$:

$$\text{BIC} = m \ln (\text{MSE}(\mathbf{Y}, \hat{\mathbf{Y}}_{\mathbf{a}})) + \|\mathbf{a}\|_0 \cdot \ln m,$$

Чем меньше значение BIC, тем лучше набор признаков.

Данные

Вычислительный эксперимент проводился на данных электрокортикограмм. Описание данных приведено в Главе 1.

На Рис. 2.2 показаны матрицы корреляций для исходных матриц \mathbf{X} и \mathbf{Y} данных ECoG. Частоты в матрице \mathbf{X} сильно коррелированы. В целевой матрице \mathbf{Y} корреляции между осями несущественны по сравнению с корреляциями между последовательными моментами времени и эти корреляции спадают со временем.

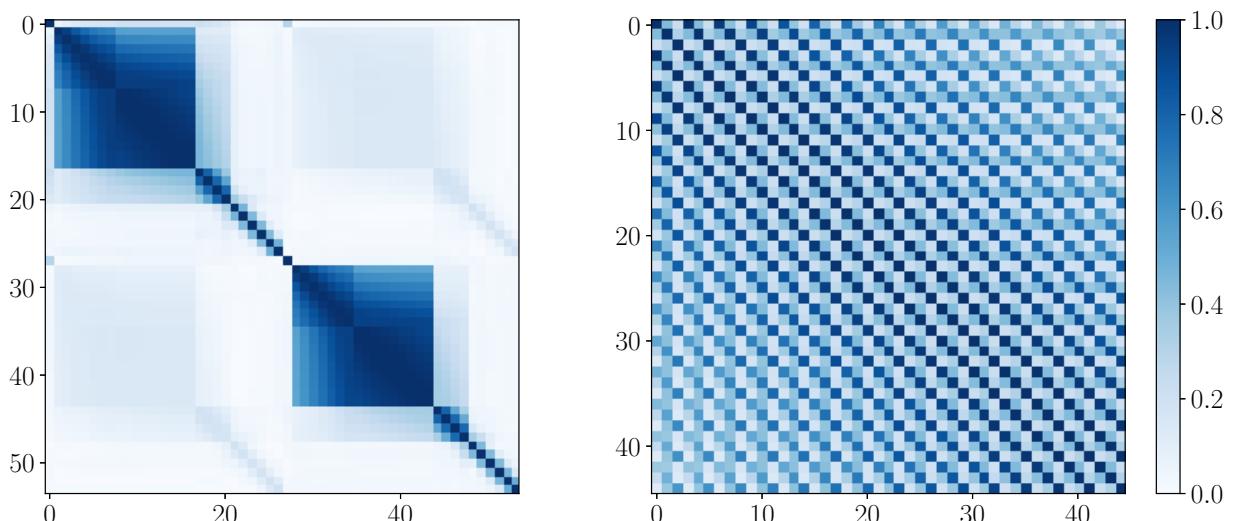


Рис. 2.2: Матрицы корреляций для матрицы плана \mathbf{X} и целевой матрицы \mathbf{Y} для данных ECoG

Результаты

Применим алгоритм SymImp QPFS для различных значений коэффициента α_3 согласно формуле (2.10). Зависимость значимости целевых векторов \mathbf{z}_y относительно коэффициента α_3 для различных значений k показана на Рис. 2.3. Значимости целевых векторов почти одинаковы для всех координат запястья при прогнозировании одного отсчёта времени ($k = 1$), что отражает независимость между координатами x , y и z . Для $k = 2$ и $k = 3$ значимости некоторых целевых векторов становится нулевой при увеличении α_3 . Вертикальные линии соответствуют оптимальному значению α_3 , вычисленному по (2.9). При этом значении α_3 значимости компонент \mathbf{z}_y совпадают. Таким образом, алгоритм не учитывает различия между целевыми векторами для $k = 1, 2, 3$.

Предлагаемые алгоритмы многомерного QPFS, приведенные в таблице 2.1 применяются для набора данных ECoG. Решим задачу выбора признаков для каждого из алгоритмов, чтобы получить вектора значимостей признаков. Отсортируем по убыванию признаки по значению их значимостей. Обучим линейную модель, постепенно добавляя в неё признаки. Исследуются значения описанных критериев качества при увеличении количества отобранных признаков. На Рис. 2.4 показаны результаты прогнозирования для случая прогнозирования $k = 30$ отсчётов времени. Порог значимости признаков τ обозначен цветными тиками. Пороговые значения τ для предлагаемых методов больше, чем для базового алгоритма RelAgg. Алгоритм SymImp имеет большой порог, не позволяя получить малый набор признаков. Однако алгоритм SymImp обладает наилучшей предсказательной способностью с точки зрения sRMSE на тестовых данных. Второй по качеству результат по sRMSE показал алгоритм AsymImp. Все предложенные алгоритмы достигают меньшей ошибки на тестовой выборке по сравнению с алгоритмом RelAgg. Критерий устойчивости также выше для предложенных алгоритмов. Алгоритм AsymImp показывает лучшие результаты с точки зрения качества прогнозирования и размера выбранного подмножества

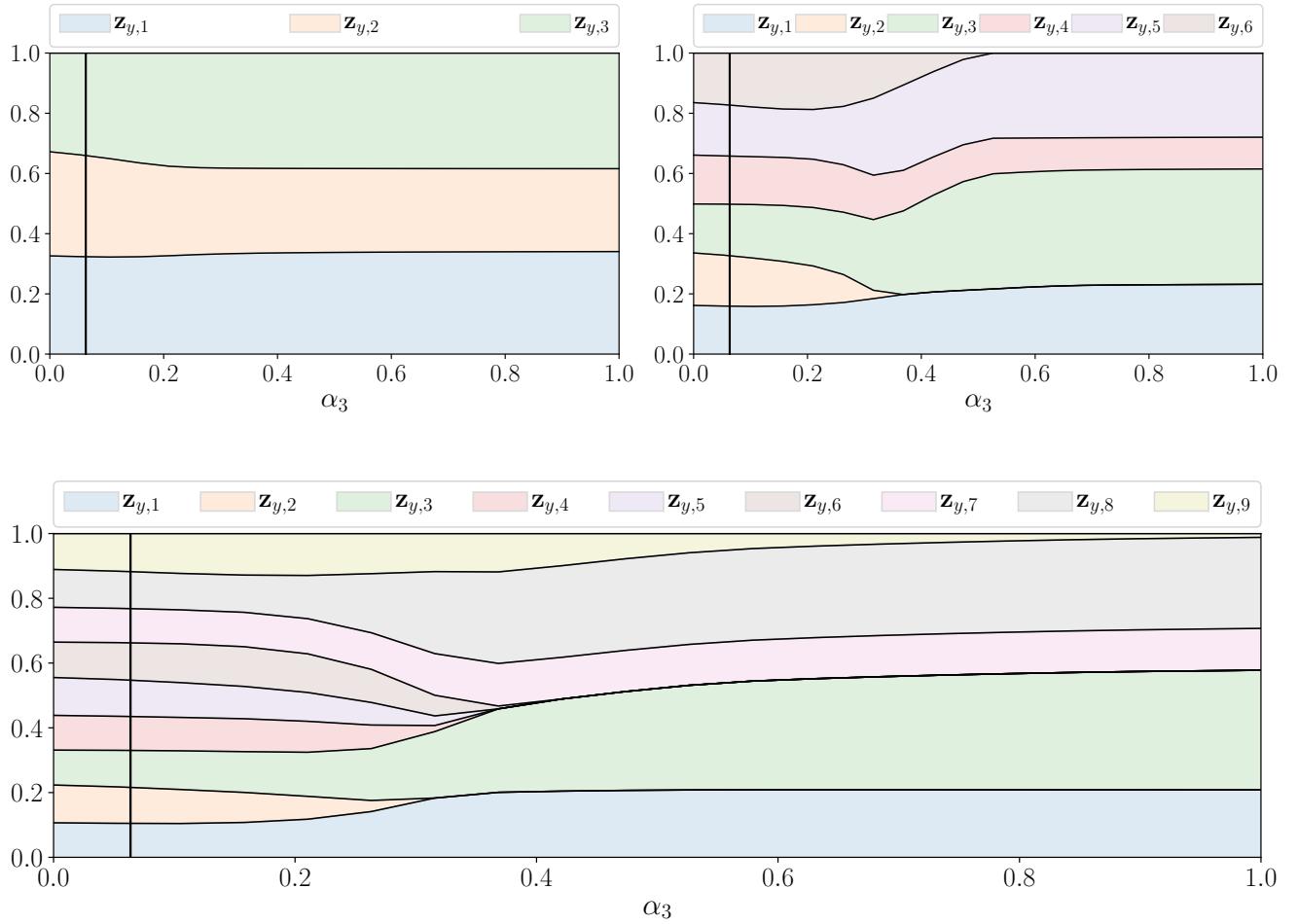


Рис. 2.3: Значимости целевых векторов \mathbf{Z}_y в зависимости от α_3 для алгоритма SymImp QPFS

признаков.

Чтобы сравнить структуру выбранных подмножеств признаков и исследовать стабильность процедуры выбора признаков, используется метод генерации данных с помощью бутстрепа. Генерируется множество подвыборок, выбирая объекты по одному с возвращениями. Затем решается задачу выбора признаков для каждой пары матрицы плана bX и целевой матрицы \mathbf{Y} . Сравниваются полученные вектора значимостей для различных подвыборок данных. В качестве меры стабильности работы алгоритмов вычисляется средний попарный коэффициент корреляции Спирмена и попарное ℓ_2 расстояние. В таблице 2.2 показана средняя ошибка sRMSE, размер подмножества признаков и описанные статистики для каждого алгоритма. Ошибка считалась на обученной линей-

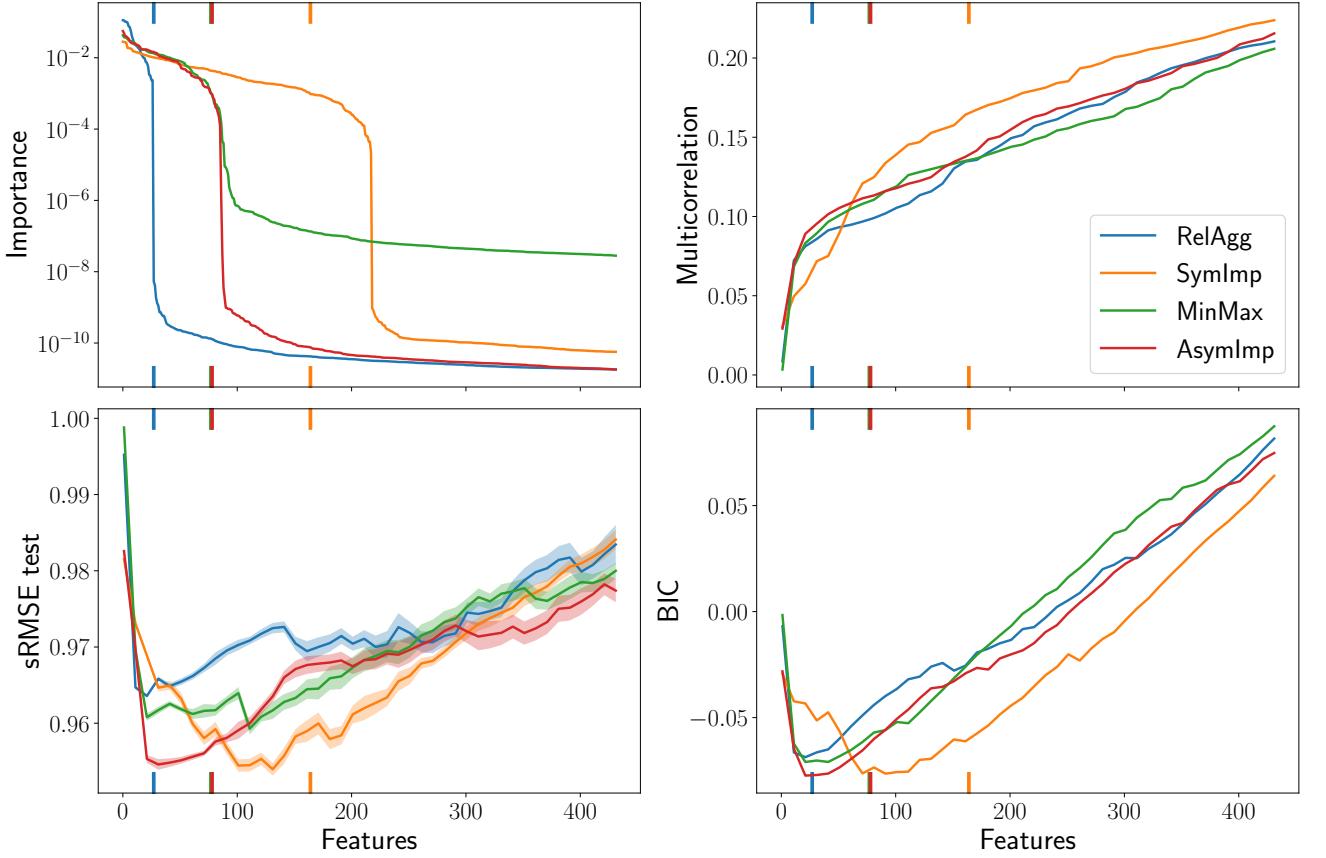


Рис. 2.4: Сравнение предложенных алгоритмов выбора признаков для данных ECoG при прогнозировании $k = 30$ отсчётов времени

ной модели с использованием 50 признаков с наибольшими значениями значимостей. Asymimp дает наименьшую ошибку на тестовой выборке. Размер выбранных подмножеств объектов завышен при использовании порогового значения $\tau = 10^{-4}$. Оптимальное значение τ может быть подобрано с помощью процедуры кросс валидации.

Для того, чтобы сравнить методы снижения размерности и выбора признаков, используется модель PLS. На Рис. 2.5 показана ошибка sRMSE на тренировочной и тестовой выборках в зависимости от размерности скрытого пространства l . Ошибка на тестовой выборке достигает минимума при $l = 11$. Алгоритм PLS является более гибким подходом по сравнению с линейной моделью, построенной на подмножестве признаков, так как использует все исходные признаки. Это приводит к меньшей ошибке, но модель не является разреженной.

Таблица 2.2: Стабильность предложенных алгоритмов выбора признаков

	sRMSE	$\ \mathbf{a}\ _0$	Spearman ρ	ℓ_2
RelAgg	0.965 ± 0.002	26.8 ± 3.8	0.915 ± 0.016	0.145 ± 0.018
SymImp	0.961 ± 0.001	224.4 ± 9.0	0.910 ± 0.017	0.025 ± 0.002
MinMax	0.961 ± 0.002	101.0 ± 2.1	0.932 ± 0.009	0.059 ± 0.004
AsymImp	0.955 ± 0.001	85.8 ± 10.2	0.926 ± 0.011	0.078 ± 0.007

На рис. 2.6 проведено сравнение 3 моделей: линейной регрессии и регрессии PLS, построенной на 100 признаках QPFS, и регрессии PLS со всеми признаками. Линейная регрессия со всеми признаками не рассматривается, так как ее результаты близки к константному прогнозу. На рисунке также приведены результаты алгоритмов lasso и elastic net, которые широко используются для выбора признаков. В данном эксперименте использовался алгоритм AsymImp QPFS. Размерность скрытого пространства PLS $l = 15$. Результаты регрессии PLS значительно лучше, линейной регрессии с признаками QPFS. Это означает, что последняя модель не является достаточно гибкой. Тем не менее, лучший результат показывает модель PLS, построенная на признаках QPFS. Данная модель является разреженной, так как использует только 100 исходных признаков. Способность модели PLS находить оптимальное скрытое представление данных улучшает предсказательную способность модели.

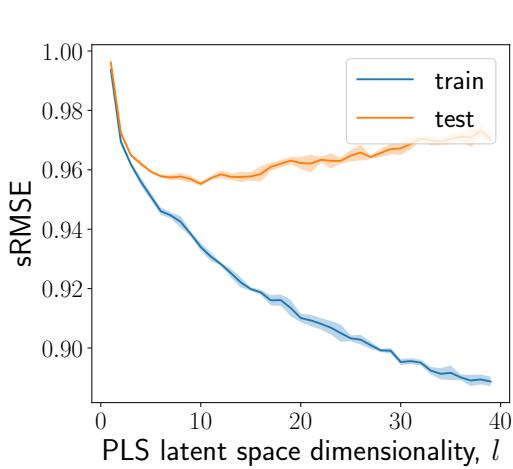


Рис. 2.5: sRMSE на тестовой выборке для модели PLS

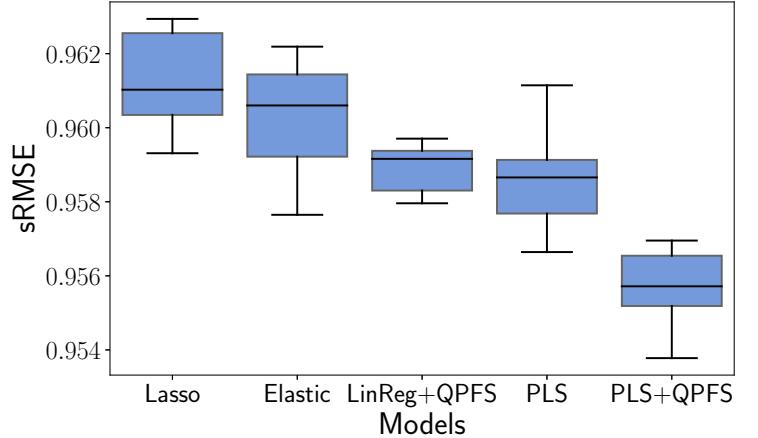


Рис. 2.6: Диаграммы размаха значений sRMSE на тестовой выборке для рассматриваемых моделей

Глава 3

Выбор параметров нелинейных моделей

3.1. Выбор параметров для обучения моделей

Модель $f(\mathbf{x}, \boldsymbol{\theta})$ с параметрами $\boldsymbol{\theta} \in \mathbb{R}^p$ предсказывает целевую переменную $y \in \mathbb{Y}$ по объекту $\mathbf{x} \in \mathbb{R}^n$. Пространство \mathbb{Y} представляет собой бинарные метки классов $\{0, 1\}$ для задачи двухклассовой классификации и \mathbb{R} для задачи регрессии. Даны матрица плана $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_m]^\top \in \mathbb{R}^{m \times n}$ и целевой вектор $\mathbf{y} = [y_1, \dots, y_m]^\top \in \mathbb{Y}^m$. Цель состоит в нахождении оптимальных параметров $\boldsymbol{\theta}^*$. Параметры $\boldsymbol{\theta}$ вычисляются минимизацией функции ошибки:

$$\boldsymbol{\theta}^* = \arg \min_{\boldsymbol{\theta} \in \mathbb{R}^p} \mathcal{L}(\boldsymbol{\theta}, \mathbf{X}, \mathbf{y}). \quad (3.1)$$

В качестве функции ошибки $\mathcal{L}(\boldsymbol{\theta}, \mathbf{X}, \mathbf{y})$ рассматриваются квадратичная ошибка для задачи регрессии:

$$\mathcal{L}(\boldsymbol{\theta}, \mathbf{X}, \mathbf{y}) = \frac{1}{2} \|\mathbf{y} - \mathbf{f}(\mathbf{X}, \boldsymbol{\theta})\|_2^2 = \frac{1}{2} \sum_{i=1}^m (y_i - f(\mathbf{x}_i, \boldsymbol{\theta}))^2, \quad (3.2)$$

и функция кросс-энтропии для задачи бинарной классификации:

$$\mathcal{L}(\boldsymbol{\theta}, \mathbf{X}, \mathbf{y}) = \sum_{i=1}^m [y_i \log f(\mathbf{x}_i, \boldsymbol{\theta}) + (1 - y_i) \log(1 - f(\mathbf{x}_i, \boldsymbol{\theta}))]. \quad (3.3)$$

Задача (3.1) решается с помощью итеративной процедуры оптимизации. Для получения параметров на шаге k текущие параметры $\boldsymbol{\theta}^{k-1}$ обновляются по следующему правилу:

$$\boldsymbol{\theta}^k = \boldsymbol{\theta}^{k-1} + \Delta\boldsymbol{\theta}^{k-1}. \quad (3.4)$$

Авторы используют метод оптимизации Ньютона для выбора вектора обновлений $\Delta\boldsymbol{\theta}$.

Метод Ньютона нестабилен и вычислительно сложен. В данной статье предлагается стабильный алгоритм Ньютона. Перед шагом градиента предлагается выбрать подмножество активных параметров модели, которые оказывают наибольшее влияние на функцию ошибки $\mathcal{L}(\boldsymbol{\theta}, \mathbf{X}, \mathbf{y})$. Обновление параметров производится только для отобранного множества индексов $\mathcal{A} = \{j : a_j = 1, \mathbf{a} \in \{0, 1\}^p\}$

$$\begin{aligned} \boldsymbol{\theta}_{\mathcal{A}}^k &= \boldsymbol{\theta}_{\mathcal{A}}^{k-1} + \Delta\boldsymbol{\theta}_{\mathcal{A}}^{k-1}, \quad \boldsymbol{\theta}_{\mathcal{A}} = \{\theta_j : j \in \mathcal{A}\}; \\ \boldsymbol{\theta}_{\bar{\mathcal{A}}}^k &= \mathbf{w}_{\bar{\mathcal{A}}}^{k-1}, \quad \boldsymbol{\theta}_{\bar{\mathcal{A}}} = \{\theta_j : j \notin \mathcal{A}\}. \end{aligned}$$

Чтобы выбрать оптимальное подмножество индексов \mathcal{A} , из всех возможных $2^p - 1$ подмножеств, вводится функция ошибки

$$\mathbf{a} = \arg \min_{\mathbf{a}' \in \{0, 1\}^p} S(\mathbf{a}', \mathbf{X}, \mathbf{y}, \boldsymbol{\theta}), \quad (3.5)$$

аналогичная функции ошибки (2.1) для задачи выбора признаков. Задача (3.5) решается на каждом шаге k процесса оптимизации для текущих параметров $\boldsymbol{\theta}^k$.

Алгоритм QPFS используется для решения задачи (3.5). QPFS выбирает подмножество параметров \mathbf{a} для вектора обновлений $\Delta\boldsymbol{\theta}$, которые оказывают наибольшее влияние на вектор остатков и являются попарно независимыми.

Функция ошибки (2.5) соответствует функции ошибки $S(\mathbf{a}, \mathbf{X}, \mathbf{y}, \boldsymbol{\theta})$

$$\mathbf{a} = \arg \max_{\mathbf{a}' \in \{1,0\}^p} S(\mathbf{a}', \mathbf{X}, \mathbf{y}, \boldsymbol{\theta}) \Leftrightarrow \arg \min_{\mathbf{a} \in \mathbb{R}_+^p, \|\mathbf{a}\|_1=1} [\mathbf{a}^\top \mathbf{Q} \mathbf{a} - \alpha \cdot \mathbf{b}^\top \mathbf{a}]. \quad (3.6)$$

В работе показано, что для модели нелинейной регрессии с квадратичной функцией ошибки (3.2) и для модели логистической регрессии с крос-энтропией (3.3), каждый шаг оптимизации эквивалентен задаче линейной регрессии (2.4).

3.2. Метод Ньютона решения задачи настройки параметров

Метод Ньютона использует условие оптимизации первого порядка для задачи (3.1) и линеаризует градиент $S(\boldsymbol{\theta})$

$$\nabla S(\boldsymbol{\theta} + \Delta \boldsymbol{\theta}) = \nabla S(\boldsymbol{\theta}) + \mathbf{H} \cdot \Delta \boldsymbol{\theta} = 0,$$

$$\Delta \boldsymbol{\theta} = -\mathbf{H}^{-1} \nabla S(\boldsymbol{\theta}).$$

где $\mathbf{H} = \nabla^2 S(\boldsymbol{\theta})$ является Гессианом матрицы функции ошибки $S(\boldsymbol{\theta})$.

Итерация (3.4) метода Ньютона имеет вид

$$\boldsymbol{\theta}^k = \boldsymbol{\theta}^{k-1} - \mathbf{H}^{-1} \nabla S(\boldsymbol{\theta}).$$

Каждая итерация инвертирует матрицу Гессиана. Мерой плохой обусловленности для матрицы Гессиана \mathbf{H} является число обусловленности

$$\varkappa(\mathbf{H}) = \frac{\lambda_{\max}(\mathbf{H})}{\lambda_{\min}(\mathbf{H})},$$

где $\lambda_{\max}(\mathbf{H}), \lambda_{\min}(\mathbf{H})$ являются максимальным и минимальным собственными значениями \mathbf{H} . Большое число обусловленности $\varkappa(\mathbf{H})$ приводит к нестабильности процесса оптимизации. Предложенный алгоритм уменьшает размер матрицы Гессиана \mathbf{H} . В наших экспериментах это приводит к меньшему числу обусловленности $\varkappa(\mathbf{H})$.

Размер шага метода Ньютона может быть чрезмерно большим. Для управления размером шага обновлений добавим параметр η в правило обновления (3.4)

$$\boldsymbol{\theta}^k = \boldsymbol{\theta}^{k-1} + \eta \Delta \boldsymbol{\theta}^{k-1}, \quad \eta \in [0, 1].$$

Для выбора соответствующего размера шага η используется правило Армихо. Выбирается максимальное η так, чтобы выполнялось следующее условие

$$S(\boldsymbol{\theta}^{k-1} + \eta \Delta \boldsymbol{\theta}^{k-1}) < S(\boldsymbol{\theta}^{k-1}) + \gamma \eta \nabla S^\top(\boldsymbol{\theta}^{k-1}) \Delta \boldsymbol{\theta}^{k-1}, \quad \gamma \in [0, 0.5].$$

Модель нелинейной регрессии

Предположим, что модель $f(\mathbf{x}, \boldsymbol{\theta})$ близка к линейной в окрестности точки $\boldsymbol{\theta} + \Delta \boldsymbol{\theta}$

$$\mathbf{f}(\mathbf{X}, \boldsymbol{\theta} + \Delta \boldsymbol{\theta}) \approx \mathbf{f}(\mathbf{X}, \boldsymbol{\theta}) + \mathbf{J} \cdot \Delta \boldsymbol{\theta},$$

где $\mathbf{J} \in \mathbb{R}^{m \times p}$ является матрицы Якоби

$$\mathbf{J} = \begin{pmatrix} \frac{\partial f(\mathbf{x}_1, \boldsymbol{\theta})}{\partial \theta_1} & \dots & \frac{\partial f(\mathbf{x}_1, \boldsymbol{\theta})}{\partial \theta_p} \\ \dots & \dots & \dots \\ \frac{\partial f(\mathbf{x}_m, \boldsymbol{\theta})}{\partial \theta_1} & \dots & \frac{\partial f(\mathbf{x}_m, \boldsymbol{\theta})}{\partial \theta_p} \end{pmatrix}. \quad (3.7)$$

В соответствии с этим предположением градиент $\nabla S(\boldsymbol{\theta})$ и Гессиан матрицы \mathbf{H} функции ошибки (3.2) равняются

$$\nabla S(\boldsymbol{\theta}) = \mathbf{J}^\top (\mathbf{y} - \mathbf{f}), \quad \mathbf{H} = \mathbf{J}^\top \mathbf{J}. \quad (3.8)$$

Это приводит к методу Гаусса-Ньютона и правилу обновления (3.4)

$$\boldsymbol{\theta}^k = \boldsymbol{\theta}^{k-1} + (\mathbf{J}^\top \mathbf{J})^{-1} \mathbf{J}^\top (\mathbf{f} - \mathbf{y}).$$

Вектор обновления $\Delta \boldsymbol{\theta}$ является решением задачи линейной регрессии

$$\|\mathbf{e} - \mathbf{F} \Delta \boldsymbol{\theta}\|_2^2 \rightarrow \min_{\Delta \boldsymbol{\theta} \in \mathbb{R}^p}, \quad (3.9)$$

где $\mathbf{e} = \mathbf{f} - \mathbf{y}$ и $\mathbf{F} = \mathbf{J}$.

В качестве нелинейной модели рассматривается модель двухслойной нейронной сети. В этом случае модель $f(\mathbf{x}, \boldsymbol{\theta})$ задается следующим образом:

$$f(\mathbf{x}, \boldsymbol{\theta}) = \sigma(\mathbf{x}^\top \mathbf{W}_1) \mathbf{w}_2.$$

Здесь $\mathbf{W}_1 \in \mathbb{R}^{m \times h}$ – это матрица весов, которые соединяют исходные признаки с h скрытыми нейронами. Функция нелинейности $\sigma(\cdot)$ применяется поэлементно. Веса $\mathbf{w}_2 \in \mathbb{R}^{h \times 1}$ соединяют скрытые нейроны с выходом. Вектор параметров модели $\boldsymbol{\theta}$ представляет собой объединение векторизованных матриц $\mathbf{W}_1, \mathbf{w}_2$.

Модель логистической регрессии

Для логистической регрессии модель имеет вид $f(\mathbf{x}, \boldsymbol{\theta}) = \sigma(\mathbf{x}^\top \boldsymbol{\theta})$ с сигмоидной функцией активации $\sigma(\cdot)$. Градиент и Гессиан функции ошибки (3.3) равны

$$\nabla S(\boldsymbol{\theta}) = \mathbf{X}^\top (\mathbf{f} - \mathbf{y}), \quad \mathbf{H} = \mathbf{X}^\top \mathbf{R} \mathbf{X}, \quad (3.10)$$

где \mathbf{R} – это диагональная матрица с диагональными элементами $f(\mathbf{x}_i, \boldsymbol{\theta}) \cdot (1 - f(\mathbf{x}_i, \boldsymbol{\theta}))$.

Правило обновления (3.4) в этом случае

$$\boldsymbol{\theta}^k = \boldsymbol{\theta}^{k-1} + (\mathbf{X}^\top \mathbf{R} \mathbf{X})^{-1} \mathbf{X}^\top (\mathbf{y} - \mathbf{f}).$$

Этот алгоритм известен как итеративный алгоритм взвешенных наименьших квадратов (IRLS). Вектор обновлений $\Delta \mathbf{w}$ является решением задачи линейной регрессии

$$\|\mathbf{e} - \mathbf{F} \Delta \boldsymbol{\theta}\|_2^2 \rightarrow \min_{\Delta \boldsymbol{\theta} \in \mathbb{R}^p}, \quad (3.11)$$

где $\mathbf{e} = \mathbf{R}^{-1/2}(\mathbf{y} - \mathbf{f})$ и $\mathbf{F} = \mathbf{R}^{1/2} \mathbf{X}$.

3.3. Метод Ньютона с выбором параметром с помощью квадратичного программирования

Предлагается реализовать алгоритм QPFS для решения задач (3.9) и (3.11). QPFS матрица \mathbf{Q} и вектор \mathbf{b} имеют вид

$$\mathbf{Q} = \text{Sim}(\mathbf{F}), \quad \mathbf{b} = \text{Rel}(\mathbf{F}, \mathbf{e}).$$

Выборочный коэффициент корреляции равен нулю для ортогональных векторов. Покажем, что в оптимальной точке $\boldsymbol{\theta}^*$ вектор \mathbf{e} ортогонален столбцам матрицы \mathbf{F} . В этом случае вектор $\mathbf{b} = \text{Rel}(\mathbf{F}, \mathbf{e})$ равен нулю. Это означает, что член, учитывающий релевантность, в данном случае исключается. Условие оптимизации первого порядка гарантирует это свойство для модели нелинейной регрессии

$$\mathbf{F}^\top \mathbf{e} = \mathbf{J}^\top (\mathbf{f} - \mathbf{y}) = -\nabla S(\boldsymbol{\theta}^*) = \mathbf{0},$$

и для модели логистической регрессии

$$\mathbf{F}^\top \mathbf{e} = \mathbf{X} \mathbf{R}^{-1/2} \mathbf{R}^{1/2} (\mathbf{y} - \mathbf{f}) = \mathbf{X}^\top (\mathbf{y} - \mathbf{f}) = \nabla S(\boldsymbol{\theta}^*) = \mathbf{0}.$$

Псевдокод предлагаемого алгоритма приведён в алгоритме 2.

3.4. Вычислительный эксперимент

Целью вычислительного эксперимента является исследование свойств предложенного алгоритма и сравнение его с другими методами.

Исследована зависимость параметров алгоритма QPFS для задач (3.9), (3.11). Предположим, что вектор параметров $\boldsymbol{\theta}^0$ лежит вблизи оптимального вектора параметров $\boldsymbol{\theta}^*$. Рассмотрим отрезок

$$\boldsymbol{\theta}_\beta = \beta \boldsymbol{\theta}^* + (1 - \beta) \boldsymbol{\theta}^0; \quad \beta \in [0, 1].$$

Сгенерируем синтетический набор данных с 300 объектами и 7 признаками для задачи логистической регрессии. Ландшафт функции ошибки (3.3) на

Algorithm 2 QPFS + Ньютон алгоритм

Вход: ε – допустимое отклонение;

τ – пороговое значение;

γ – параметр правила Армихо.

Выход: $\boldsymbol{\theta}^*$;

инициализировать $\boldsymbol{\theta}^0$;

$k := 1$;

повторять

вычислить \mathbf{e} и \mathbf{F} для (3.9) или (3.11) ;

$\mathbf{Q} := \text{Sim}(\mathbf{F})$, $\mathbf{b} := \text{Rel}(\mathbf{F}, \mathbf{e})$, $\alpha = \frac{\bar{\mathbf{Q}}}{\bar{\mathbf{Q}} + \bar{\mathbf{b}}}$;

$\mathbf{a} := \arg \min_{\mathbf{a} \geq 0, \|\mathbf{a}\|_1=1} \mathbf{a}^\top \mathbf{Q} \mathbf{a} - \alpha \cdot \mathbf{b}^\top \mathbf{a}$;

$\mathcal{A} := \{j : a_j = 1\}$;

вычислить $\nabla S(\boldsymbol{\theta}^{k-1})$, \mathbf{H} для (3.8) или (3.10);

$\Delta \boldsymbol{\theta}^{k-1} = -\mathbf{H}^{-1} \nabla S(\boldsymbol{\theta}^{k-1})$;

$\eta := \text{ArmijoRule}(\boldsymbol{\theta}^{k-1}, \gamma)$;

$\boldsymbol{\theta}_{\mathcal{A}}^k = \boldsymbol{\theta}_{\mathcal{A}}^{k-1} + \eta \Delta \boldsymbol{\theta}_{\mathcal{A}}^{k-1}$;

$k := k + 1$;

пока $\frac{\|\boldsymbol{\theta}^k - \boldsymbol{\theta}^{k-1}\|}{\|\boldsymbol{\theta}^k\|} < \varepsilon$

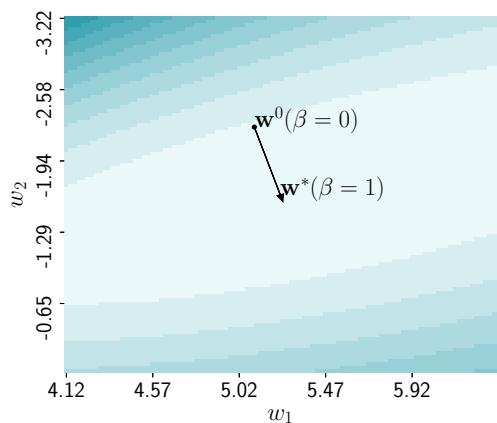


Рис. 3.1: Ландшафт функции ошибки для логистической регрессии

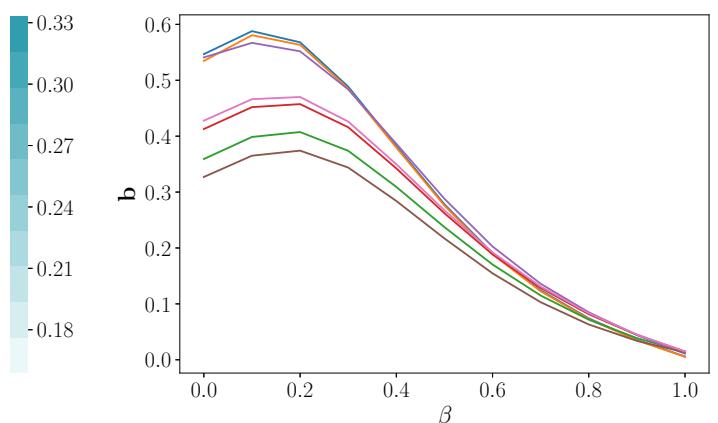


Рис. 3.2: Релевантности параметров для логистической регрессии

сетке двух случайно выбранных параметров показан на рис. 3.1. Поверхность функции ошибки выпуклая с вытянутыми линиями уровня вдоль некоторых параметров модели. Добавим случайный шум к оптимальным параметрам θ^* , чтобы получить точку θ^0 . Поведение вектора \mathbf{b} на отрезке между θ^0 и θ^* показано на рис. 3.2. Компоненты \mathbf{b} начинают резко уменьшаться, приближаясь к оптимальной точке.

Для модели нелинейной регрессии используется классический набор данных Boston Housing с 506 объектами и 13 признаками. Для простоты нейронная сеть содержит два скрытых нейрона. Ландшафт функции ошибок для модели нейронной сети является более сложным. Он не выпуклый и может содержать несколько локальных минимумов. Двумерный ландшафт функции ошибок для этого набора данных показан на рис. 3.3. Сетка строится для двух случайных весов из матрицы \mathbf{W}_1 . Мы используем ту же стратегию для исследования того, как вектор \mathbf{b} изменяется от θ^0 до θ^* . Результат показан на рис. 3.4. Компоненты вектора \mathbf{b} становятся близки к нулю вблизи оптимума. При достижении оптимального значения различные веса влияют на остатки модели \mathbf{e} .

На рис. 3.5 показан процесс оптимизации для предложенного алгоритма в случае логистической регрессии с двумя параметрами модели. Даже для двумерной задачи решение метода Ньютона нестабильно и число обусловленности

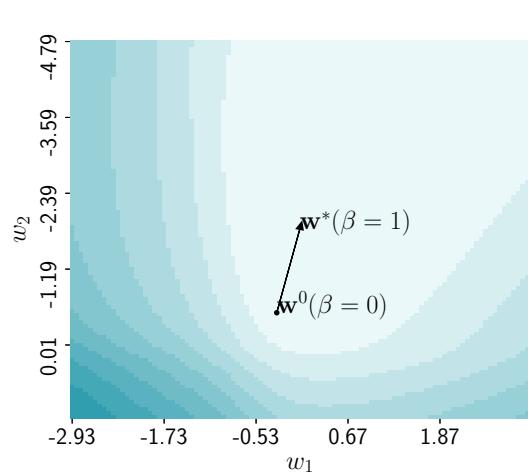


Рис. 3.3: Ландшафт функции ошибки для нейронной сети

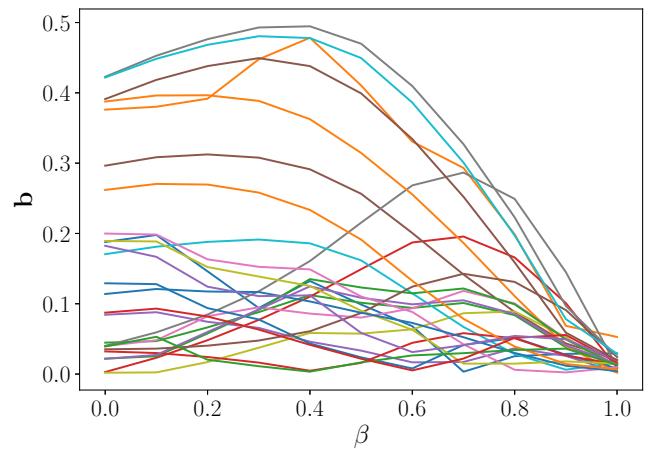


Рис. 3.4: Релевантности параметров первого слоя для модели нейронной сети

матрицы Гессиана \mathbf{H} может быть чрезвычайно большим. На каждом шаге алгоритма процедура QPFS выбирает параметры для оптимизации. В данном примере предложенный алгоритм выбирает и обновляет только один параметр на каждой итерации на первых шагах. Это делает алгоритм более устойчивым.

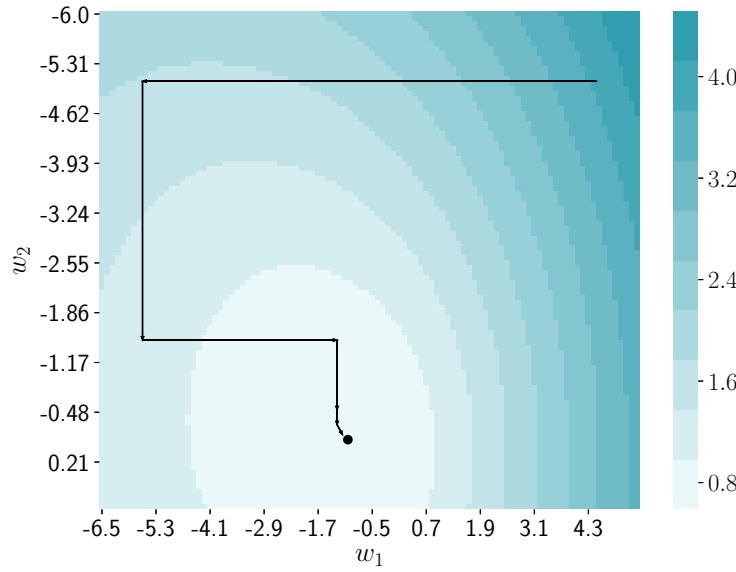


Рис. 3.5: Оптимизационный процесс предложенного алгоритма QPFS+Ньютон для модели логистической регрессии

На рис. 3.6 показаны наборы активных параметров на итерациях для набора

данных Boston Housing и нейронной сети с двумя скрытыми нейронами. Темные ячейки соответствуют активным параметрам, которые мы оптимизируем.

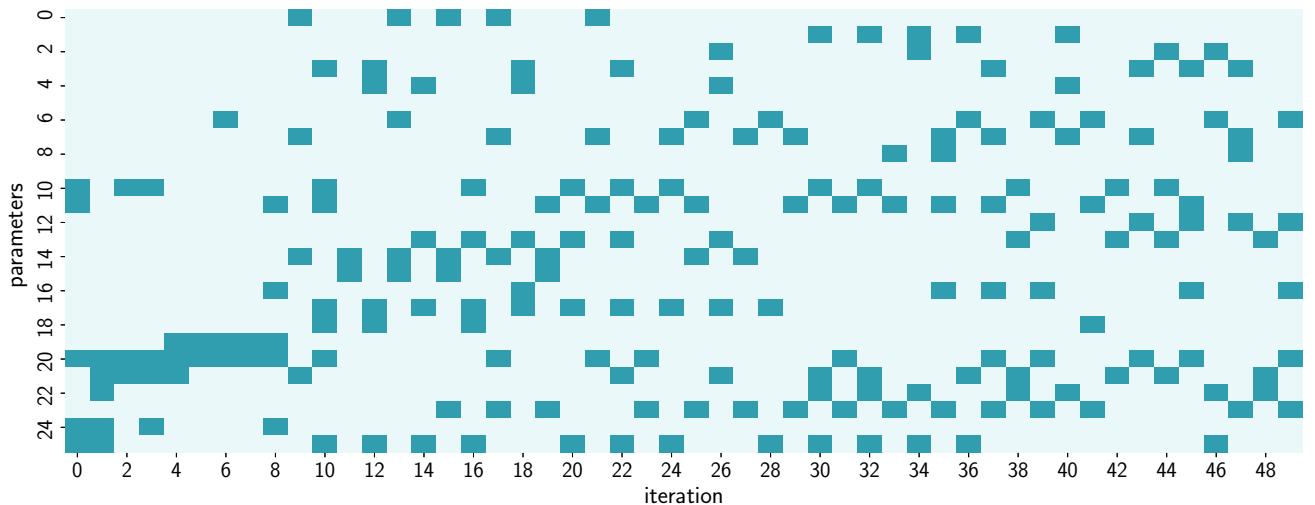


Рис. 3.6: Множества активных параметров на протяжении оптимизационного процесса

В рассмотренных примерах число обусловленности $\kappa(\mathbf{H})$ для метода Ньютона на некоторых итерациях было чрезвычайно большим. Выбор активных параметров позволил значительно сократить число обусловленности.

Мы сравнили предложенный алгоритм с существующими методами, а именно градиентным спуском (GD), моментом Нестерова, Adam и оригинальным алгоритмом Ньютона. Проведены эксперименты для моделей нелинейной и логистической регрессий. Наборы данных были выбраны из репозитория UCI [65]. Результаты показаны в таблицах 3.1 и 3.2. Для каждого набора данных две строки содержат ошибки для тренировочной (первая строка) и тестовой (вторая строка) выборок. В таблице 3.1 приведена квадратичная ошибка, в таблице 3.2 – кросс-энтропия. Чтобы найти среднюю ошибку и ее стандартное отклонение использовалась процедура кросс валидации на 5 фолдов. Предложенный алгоритм показывает меньшую ошибку на трех из четырех наборов данных для нелинейной регрессии и среди двух из трех наборов данных для логистической регрессии.

Таблица 3.1: Средняя квадратичная ошибка на тренировочной и тестовой выборках для модели нелинейной регрессии

Выборка	$\frac{m}{n}$	GD	Нестеров	ADAM	Ньютон	QPFS+Ньютон
Boston House Prices	506	27.2 ± 4.6	46.0 ± 11.0	35.4 ± 2.5	22.1 ± 15.2	20.9 ± 10.4
	13	32.4 ± 5.6	53.3 ± 11.5	37.8 ± 7.0	28.9 ± 13.6	24.5 ± 9.4
Communities and Crime	1994	48.0 ± 6.4	31.4 ± 2.8	23.3 ± 3.7	18.3 ± 3.4	26.7 ± 3.1
	99	47.5 ± 6.5	32.9 ± 4.3	28.1 ± 4.5	28.8 ± 3.6	28.4 ± 3.0
Forest Fires	517	18.9 ± 0.4	1.83 ± 0.4	1.81 ± 0.6	17.7 ± 0.4	17.9 ± 0.4
	10	20.0 ± 2.1	20.2 ± 2.2	20.0 ± 2.0	20.6 ± 1.4	20.2 ± 2.2
Residential Building	372	51.6 ± 17.7	32.6 ± 19.5	30.0 ± 24.8	35.5 ± 24.7	30.3 ± 10.7
	103	53.7 ± 13.9	34.1 ± 13.6	34.1 ± 19.4	35.0 ± 15.6	30.9 ± 5.3

Таблица 3.2: Среднее значение кросс-энтропии на тренировочной и тестовой выборках для модели логистической регрессии

Выборка	$\frac{m}{n}$	GD	Нестеров	ADAM	Ньютон	QPFS+Ньютон
Breast Cancer	569	0.6 ± 0.1	0.4 ± 0.1	0.8 ± 0.2	0.3 ± 0.1	0.2 ± 0.1
	30	0.9 ± 0.2	1.0 ± 0.7	1.2 ± 0.2	1.0 ± 0.2	1.1 ± 0.3
Cardiotocography	2126	11.5 ± 4.7	11.5 ± 4.7	8.8 ± 4.4	11.5 ± 5.7	7.7 ± 4.2
	21	11.6 ± 5.8	11.5 ± 5.7	9.0 ± 2.6	11.5 ± 4.7	7.7 ± 4.7
Climate Model Simulation Crashes	540	1.2 ± 0.1	1.0 ± 0.2	1.5 ± 0.2	1.0 ± 0.5	0.8 ± 0.3
	18	1.4 ± 2.0	1.3 ± 0.7	1.8 ± 0.3	1.2 ± 0.5	1.1 ± 0.4

Глава 4

Метрические методы

[связать все определения](#)

4.1. Метрическое обучение в задачах кластеризации временных рядов

Пусть $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_m]^T \in \mathbb{R}^{m \times n}$ — матрица плана. Объект $\mathbf{x}_i = [x_i^1, \dots, x_i^n]^T$ задан в виде вектора в пространстве признаков. Требуется выявить кластерную

структуре данных и разбить множество объектов \mathbf{X} на множество непересекающихся кластеров, т. е. построить отображение

$$f : \mathbf{X} \rightarrow \{1, \dots, r\}.$$

Обозначим $y_i = f(\mathbf{x}_i)$, $y_i \in \{1, \dots, r\}$, — метка кластера объекта \mathbf{x}_i . Необходимо выбрать метки кластеров $\{y_i\}_{i=1}^m$ таким образом, чтобы расстояния между кластерами были максимальными. Центроиды $\boldsymbol{\mu}$ множества объектов \mathbf{X} и центроиды кластеров $\{\boldsymbol{\mu}_j\}_{j=1}^r$ вычисляются по формулам:

$$\boldsymbol{\mu} = \frac{1}{m} \sum_{i=1}^m \mathbf{x}_i; \quad \boldsymbol{\mu}_j = \frac{\sum_{i=1}^m [y_i = y_j] \mathbf{x}_i}{\sum_{i=1}^m [y_i = y_j]}. \quad (4.1)$$

Введем на множестве объектов \mathbf{X} расстояние Махаланобиса

$$\rho_{\mathbf{A}}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^{\top} \mathbf{A}^{-1} (\mathbf{x}_i - \mathbf{x}_j)}, \quad (4.2)$$

где \mathbf{A} — это матрица ковариаций множества \mathbf{X}

$$\mathbf{A} = \frac{1}{m} \sum_{i=1}^m (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^{\top}. \quad (4.3)$$

Определение 11. Функцией ошибки кластеризации назовем межкластерное расстояние:

$$\mathcal{L}(\{\boldsymbol{\mu}_j\}_{j=1}^r, \mathbf{X}, \mathbf{y}) = - \sum_{j=1}^r N_j \rho_{\mathbf{A}}^2(\boldsymbol{\mu}_j, \boldsymbol{\mu}), \quad (4.4)$$

где $N_j = \sum_{i=1}^m [y_i = y_j]$ — число объектов в кластере j .

Поставим задачу кластеризации как задачу минимизации функции ошибки (4.4)

$$(\{\boldsymbol{\mu}_j\}_{j=1}^r, \mathbf{X}, \mathbf{y}) \rightarrow \min_{\boldsymbol{\mu}_j \in \mathbb{R}^n}. \quad (4.5)$$

Для решения этой задачи предлагается применить метод метрического обучения к ковариационной матрице \mathbf{A} . Найдем такую матрицу \mathbf{A} , для которой функционал качества принимает максимальное значение:

$$\mathbf{A}^* = \arg \min_{\mathbf{A} \in \mathbb{R}^{n \times n}} S(\{\boldsymbol{\mu}_j^*\}_{j=1}^r, \mathbf{X}, \mathbf{y}), \quad (4.6)$$

где $\{\boldsymbol{\mu}_j^*\}_{j=1}^r$ — решение задачи кластеризации (4.5).

4.2. Алгоритм адаптивного метрического обучения

Для решения задач (4.5), (4.6) используется алгоритм адаптивного метрического обучения. Предлагается понизить размерность пространства объектов \mathbf{X} с помощью линейного ортогонального преобразования $\mathbf{P} \in \mathbb{R}^{n \times l}$, $\mathbf{P}^T \mathbf{P} = \mathbf{I}$, где новая размерность $l < n$

$$\mathbf{t}_i = \mathbf{P} \mathbf{x}_i \in \mathbb{R}^l, \quad i = 1, \dots, m.$$

Центроид $\hat{\boldsymbol{\mu}}$ множества объектов $\{\mathbf{t}_i\}_{i=1}^m$ вычисляется по формуле (4.1). Расстояния между объектами вычисляются по формуле (4.2), где в качестве матрицы $\hat{\mathbf{A}}$ используется матрица ковариаций (4.3) множества объектов $\{\hat{\mathbf{x}}_i\}_{i=1}^m$

$$\hat{\mathbf{A}} = \frac{1}{m} \sum_{i=1}^m (\mathbf{t}_i - \hat{\boldsymbol{\mu}})(\mathbf{t}_i - \hat{\boldsymbol{\mu}})^T = \frac{1}{m} \sum_{i=1}^m \mathbf{P}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T \mathbf{P}^T = \mathbf{P} \mathbf{A} \mathbf{P}^T.$$

Определение 12. Индикаторной матрицей назовем матрицу $\mathbf{Y} = \{y_{ij}\} \in \mathbb{R}^{m \times r}$, где

$$y_{ij} = \begin{cases} 1, & \text{если } f(\mathbf{x}_i) = y_j; \\ 0, & \text{если } f(\mathbf{x}_i) \neq y_j. \end{cases}$$

Определение 13. Взвешенной индикаторной матрицей назовем матрицу $\mathbf{L} = \mathbf{Y}(\mathbf{Y}^T \mathbf{Y})^{-1/2} = \{l_{ij}\} \in \mathbb{R}^{m \times r}$, элементы которой равны:

$$l_{ij} = \begin{cases} \frac{1}{\sqrt{N_j}}, & \text{если } f(\mathbf{x}_i) = y_j; \\ 0, & \text{если } f(\mathbf{x}_i) \neq y_j. \end{cases}$$

Теорема 3. С использованием данных обозначений задача кластеризации (4.5) и задача метрического обучения (4.6) сводятся к общей задаче минимизации функции ошибки [66]

$$\mathcal{L} = -\frac{1}{m} \text{trace}(\mathbf{L}^T \mathbf{X}^T \mathbf{P}^T \hat{\mathbf{A}}^{-1} \mathbf{P} \mathbf{X} \mathbf{L}) = -\frac{1}{m} \text{trace}(\mathbf{L}^T \mathbf{X}^T \mathbf{P}^T (\mathbf{P} \mathbf{A} \mathbf{P}^T)^{-1} \mathbf{P} \mathbf{X} \mathbf{L}) \rightarrow \min_{\mathbf{P}, \mathbf{L}}. \quad (4.7)$$

Для решения задачи (4.7) используется EM алгоритм. На каждом шаге итеративно вычисляются текущие оптимальные значения матриц \mathbf{P} и \mathbf{L} . На E -шаге необходимо найти матрицу \mathbf{L} , которая является решением оптимизационной задачи (4.7) при фиксированной матрице \mathbf{P} . В качестве начального приближения получим взвешенную индикаторную матрицу \mathbf{L} с помощью алгоритма кластеризации k -средних с евклидовой метрикой. На M -шаге производится нахождение оптимального значения матрицы \mathbf{P} при фиксированной матрице \mathbf{L} . Алгоритм завершается при стабилизации функционала \mathcal{L} на последовательности итераций.

Алгоритм k -средних

В данной работе базовым алгоритмом для сравнения является алгоритм k -средних. На первом шаге алгоритм выбирает из множества \mathbf{X} случайным образом r объектов $\{\boldsymbol{\mu}_j\}_{j=1}^r$ — начальные центроиды кластеров. Для каждого объекта \mathbf{x}_i вычисляется расстояние (4.2) до каждого центроида кластера $\boldsymbol{\mu}_j$ с единичной матрицей трансформаций \mathbf{A} . Объект \mathbf{x}_i относится к кластеру, расстояние до которого оказалось наименьшим. Далее производится вычисление новых центроидов кластеров по формуле (4.1). Алгоритм завершается, если значения центроидов кластеров стабилизируются.

Оптимизация матрицы \mathbf{P} с фиксированной матрицей \mathbf{L}

Для любых двух квадратных матриц \mathbf{A} и \mathbf{B} справедливо $\text{trace}(\mathbf{AB}) = \text{trace}(\mathbf{BA})$. Данное свойство позволяет переформулировать задачу (4.7) следующим образом:

$$\mathcal{L} = -\frac{1}{m} \text{trace}(\mathbf{L}^\top \mathbf{X}^\top \mathbf{P}^\top (\mathbf{P}\mathbf{A}\mathbf{P}^\top)^{-1} \mathbf{P}\mathbf{XL}) = -\frac{1}{m} \text{trace}((\mathbf{P}\mathbf{A}\mathbf{P}^\top)^{-1} \mathbf{P}\mathbf{XL}\mathbf{L}^\top \mathbf{X}^\top \mathbf{P}^\top).$$

Теорема 4. Обозначим $\mathbf{B} = \mathbf{XL}\mathbf{L}^\top \mathbf{X}^\top$. Обозначим через $\mathbf{P} = [\mathbf{v}_1, \dots, \mathbf{v}_r]^\mathbf{T}$ матрицу, состоящую из r собственных векторов матрицы $\mathbf{A}^{-1}\mathbf{B}$, отвечающих наи-

большим собственным значениям. Тогда решением (4.7) является ортогональная матрица, полученная QR -разложением матрицы \mathbf{P}^\top .

Доказательство. Функция ошибки \mathcal{L} зависит только от матрицы \mathbf{P} . Обозначим

$$s(\mathbf{P}) = \text{trace}((\mathbf{P}\mathbf{A}\mathbf{P}^\top)^{-1}\mathbf{P}\mathbf{B}\mathbf{P}^\top).$$

На данном шаге задача (4.7) принимает вид:

$$\mathbf{P}^* = \arg \max_{\mathbf{P} \in \mathbb{R}^{n \times l}} s(\mathbf{P}); \quad (4.8)$$

$$\mathbf{P}\mathbf{P}^\top = \mathbf{I}. \quad (4.9)$$

Ранг произведения матриц не превосходит рангов сомножителей, поэтому ранг матрицы \mathbf{B} не превосходит r . Решением (4.8) является матрица $\mathbf{P} = [\mathbf{v}_1, \dots, \mathbf{v}_r]^\top$, состоящая из r собственных векторов матрицы $\mathbf{A}^{-1}\mathbf{B}$, отвечающих наибольшим собственным значениям. Таким образом, размерность нового пространства объектов будет равна количеству кластеров r .

В общем случае матрица \mathbf{P} не является ортогональной. Заметим, что для любой невырожденной матрицы \mathbf{P} верно $s(\mathbf{P}) = s(\mathbf{M}\mathbf{P})$. Для учета условия ортогональности (4.9) найдем QR -разложение матрицы \mathbf{P} . Тогда ортогональная матрица \mathbf{Q} является оптимальным значением \mathbf{P}^* . \square

Оптимизация матрицы \mathbf{L} с фиксированной матрицей \mathbf{P}

Теорема 5. Обозначим $\hat{\mathbf{K}} = (1/N)\mathbf{X}^\top \mathbf{P}^\top \hat{\mathbf{A}}^{-1} \mathbf{P} \mathbf{X}$. Тогда задача (4.7) эквивалентна задаче кластеризации k -средних с заданным ядром $\hat{\mathbf{K}}$ [67].

При фиксированной матрице \mathbf{P} задача (4.7) принимает вид:

$$\text{trace}(\mathbf{L}^\top \hat{\mathbf{K}} \mathbf{L}) \rightarrow \max_{\mathbf{L} \in \mathbb{R}^{m \times r}} .$$

Матрица $\hat{\mathbf{K}}$ является симметричной и неотрицательно определенной, тем самым может быть выбрана в качестве ядра.

4.3. Постановка задачи

Пусть объект $\mathbf{x}_i \in \mathbb{R}^n$ — временной ряд, последовательность измерений некоторой исследуемой величины в различные моменты времени. Пусть \mathbf{X} — множество всех временных рядов фиксированной длины n , $Y = \{1, \dots, K\}$ — множество меток классов. Пусть задана выборка $\mathfrak{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^\ell$ — множество объектов с известными метками классов $y_i \in Y$.

Требуется построить точную, простую, устойчивую модель классификации

$$a : \mathbf{X} \rightarrow Y.$$

Данную модель представим в виде суперпозиции

$$a(\mathbf{x}) = b \circ \mathbf{f} \circ G(\mathbf{x}, \{\mathbf{c}_e\}_{e=1}^K), \quad (4.10)$$

где G — процедура выравнивания временных рядов относительно центроидов классов $\{\mathbf{c}_e\}_{e=1}^K$, \mathbf{f} — алгоритм метрического обучения, b — алгоритм многоклассовой классификации.

4.3.1. Выравнивание временных рядов.

Для повышения качества и устойчивости алгоритма классификации предлагается провести выравнивание временных рядов каждого класса относительно центроида.

Пусть \mathbf{X}_e — множество объектов обучающей выборки \mathfrak{D} , принадлежащих одному классу $e \in \{1, \dots, K\}$. Центроидом множества объектов $\mathbf{X}_e = \{\mathbf{x}_i | y_i = e\}_{i=1}^\ell$ по расстоянию ρ назовем вектор $\mathbf{c}_e \in \mathbb{R}^n$ такой, что

$$\mathbf{c}_e = \operatorname{argmin}_{\mathbf{c} \in \mathbb{R}^n} \sum_{\mathbf{x}_i \in \mathbf{X}_e} \rho(\mathbf{x}_i, \mathbf{c}). \quad (4.11)$$

Для нахождения центроида предлагается в качестве расстояния между временными рядами использовать путь наименьшей стоимости [?], найденный методом динамической трансформации времени. Псевдокод решения оптимизационной задачи (4.11) приведен в алгоритме 3.

Algorithm 3 Нахождение центроида DBA(\mathbf{X}_e , n_iter)

Вход: \mathbf{X}_e — множество временных рядов, принадлежащих одному и тому же классу, n_iter — количество итераций алгоритма.

Выход: \mathbf{c} — центроид множества \mathbf{X}_e .

- 1: задать начальное приближение приближение центроида \mathbf{c} ;
- 2: **для** $i = 1, \dots, n_{\text{iter}}$
- 3: **для** $\mathbf{x} \in \mathbf{X}_e$
- 4: вычислить выравнивающий путь между \mathbf{c} и \mathbf{x}
 $\text{alignment}(\mathbf{x}) := \text{DTWalignment}(\mathbf{c}, \mathbf{x});$
- 5: объединить поэлементно множества индексов для каждого отсчета времени
 $\text{alignment} := \bigcup_{\mathbf{x} \in \mathbf{X}_e} \text{alignment}(\mathbf{x});$
- 6: $\mathbf{c} = \text{mean}(\text{alignment})$

DTWalignment(\mathbf{c}, \mathbf{x})

Вход: \mathbf{c}, \mathbf{x} — временные ряды.

Выход: alignment — выравнивающий путь. // каждый индекс временного ряда \mathbf{x} поставлен в однозначное соответствие индексу временного ряда \mathbf{c}

- 1: построить $n \times n$ -матрицу деформаций DTW
 $\text{cost} := \text{DTW}(\mathbf{c}, \mathbf{x});$
 - 2: вычислить выравнивающий путь по матрице деформаций
 $\text{alignment} := \text{DTWpath}(\text{cost});$
-

Общая процедура выравнивания имеет следующий вид:

- 1) построить множество центроидов классов $\{\mathbf{c}_e\}_{e=1}^K$;
- 2) по множеству центроидов найти пути наименьшей стоимости между каждым временным рядом \mathbf{x}_i и центроидом его класса \mathbf{c}_{y_i} ;
- 3) по каждому пути восстановить выравненный временной ряд;
- 4) привести множества выравненных временных рядов к нулевому среднему и нормировать на дисперсию.

Результатом выравнивания должно стать множество выравненных временных рядов.

4.3.2. Метрическое обучение.

Введем на множестве выравненных временных рядов расстояние Махаланобиса

$$d_{\mathbf{A}}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{A} (\mathbf{x}_i - \mathbf{x}_j)},$$

где матрица трансформаций $\mathbf{A} \in \mathbb{R}^{n \times n}$ является симметричной и неотрицательно определенной ($\mathbf{A}^T = \mathbf{A}$, $\mathbf{A} \succeq 0$). Представим матрицу \mathbf{A} в виде разложения $\mathbf{A} = \mathbf{L}^T \mathbf{L}$. Матрица $\mathbf{L} \in \mathbb{R}^{p \times n}$ — матрица линейного преобразования, где p задает размерность преобразованного пространства. Если параметр $p < n$, то происходит снижение размерности признакового пространства.

Расстояние $d_{\mathbf{A}}(\mathbf{x}_i, \mathbf{x}_j)$ есть евклидово расстояние между \mathbf{Lx}_i и \mathbf{Lx}_j :

$$d_{\mathbf{A}}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{L}^T \mathbf{L} (\mathbf{x}_i - \mathbf{x}_j)} = \sqrt{(\mathbf{L}(\mathbf{x}_i - \mathbf{x}_j))^T (\mathbf{L}(\mathbf{x}_i - \mathbf{x}_j))} = \|\mathbf{L}(\mathbf{x}_i - \mathbf{x}_j)\|_2$$

В качестве алгоритма метрического обучения в данной работе был выбран алгоритм LMNN. Данный алгоритм сочетает в себе идеи метода k ближайших соседей. Первая идея заключается в минимизации расстояний между k ближайшими объектами, находящимися в одном классе. Запишем функционал качества в виде

$$Q_1(\mathbf{L}) = \sum_{j \sim i} \|\mathbf{L}(\mathbf{x}_i - \mathbf{x}_j)\|^2 \rightarrow \min_{\mathbf{L}},$$

где $j \rightsquigarrow i$ означает, что \mathbf{x}_j является одним из k ближайших соседей для \mathbf{x}_i . Вторая идея состоит в максимизации расстояния между каждым объектом и его объектами-нарушителями. Объектом-нарушителем для \mathbf{x}_i назовем объект \mathbf{x}_l такой, что

$$\|\mathbf{L}(\mathbf{x}_i - \mathbf{x}_l)\|^2 \leq \|\mathbf{L}(\mathbf{x}_i - \mathbf{x}_j)\|^2 + 1, \quad \text{где } j \rightsquigarrow i. \quad (4.12)$$

Таким образом, необходимо минимизировать следующий функционал:

$$Q_2(\mathbf{L}) = \sum_{j \rightsquigarrow i} \sum_l (1 - y_{il}) [1 + \|\mathbf{L}(\mathbf{x}_i - \mathbf{x}_j)\|^2 - \|\mathbf{L}(\mathbf{x}_i - \mathbf{x}_l)\|^2]_+ \rightarrow \min_{\mathbf{L}},$$

где $y_{il} = 1$, если $y_i = y_l$, и $y_{il} = 0$ в противном случае. Положительная срезка позволяет штрафовать только те объекты, которые удовлетворяют условию (4.12).

Задача метрического обучения состоит в нахождении линейного преобразования $\mathbf{f}(\mathbf{x}) = \mathbf{L}\mathbf{x}$, то есть нахождении матрицы \mathbf{L} в виде решения оптимизационной задачи

$$Q(\mathbf{L}) = \mu Q_1(\mathbf{L}) + (1 - \mu) Q_2(\mathbf{L}) \rightarrow \min_{\mathbf{L}}, \quad (4.13)$$

где $\mu \in (0, 1)$ — весовой параметр, определяющий вклад каждого из функционалов. Задача (4.13) представляет собой задачу полуопределенного программирования [?] и может быть решена существующими оптимизационными пакетами.

4.3.3. Классификация временных рядов.

Пусть $\mathbf{x} \in \mathbf{X}$ — неразмеченный временной ряд. Выравниваем временной ряд \mathbf{x} относительно всех центроидов классов

$$\hat{\mathbf{x}}_e = G(\mathbf{x}, \mathbf{c}_e), \quad \text{где } e = \{1, \dots, K\}.$$

Отнесем временной ряд к классу, для которого минимально расстояние до соответствующего центроида. В качестве расстояния используем обученную метрику Махalanобиса с фиксированной матрицей \mathbf{A}

$$\hat{y} = \underset{e \in \{1, \dots, K\}}{\operatorname{argmin}} d_{\mathbf{A}}(\hat{\mathbf{x}}_e, \mathbf{c}_e).$$

После нахождения оптимальных центроидов классов и нахождения оптимальной матрицы трансформаций процедура классификации заключается в измерении расстояния между найденными центроидами и новыми неразмеченными объектами.

Для оценки качества работы алгоритма будем вычислять ошибку классификации как долю неправильно классифицированных объектов тестовой выборки \mathfrak{U} :

$$\text{error} = \frac{1}{|\mathfrak{U}|} \sum_{i=1}^{|\mathfrak{U}|} [a(\mathbf{x}_i) \neq y_i].$$

4.4. Вычислительный эксперимент

В целях проверки работоспособности предложенного подхода проведен вычислительный эксперимент на модельных данных. Сгенерирована выборка объектов, принадлежащих одному из двух классов, в двумерном пространстве. Каждый объект принадлежит многомерному нормальному распределению. На рис. 1 показано истинное распределение объектов, черным цветом выделены истинные центры классов и линии уровня функции распределения.

Применим к данной выборке базовый алгоритм k -средних. Результат кластеризации показан на рис. 2, где черным цветом выделены найденные центры классов и линии уровня функции распределения, построенной по выборочной ковариационной матрице.

Взяв за начальное приближение результаты работы алгоритма k -средних, проведем кластеризацию с помощью алгоритма адаптивного метрического обучения. Результаты работы алгоритма продемонстрированы на рис. 3.

На рисунках заметно улучшение результатов кластеризации. Измеренная точность кластеризации алгоритма k -средних составила 0,76, алгоритма адаптивного метрического обучения — 0,94, что говорит о работоспособности данного подхода.

Таблица 4.1 показывает результаты вычислительного эксперимента на ре-

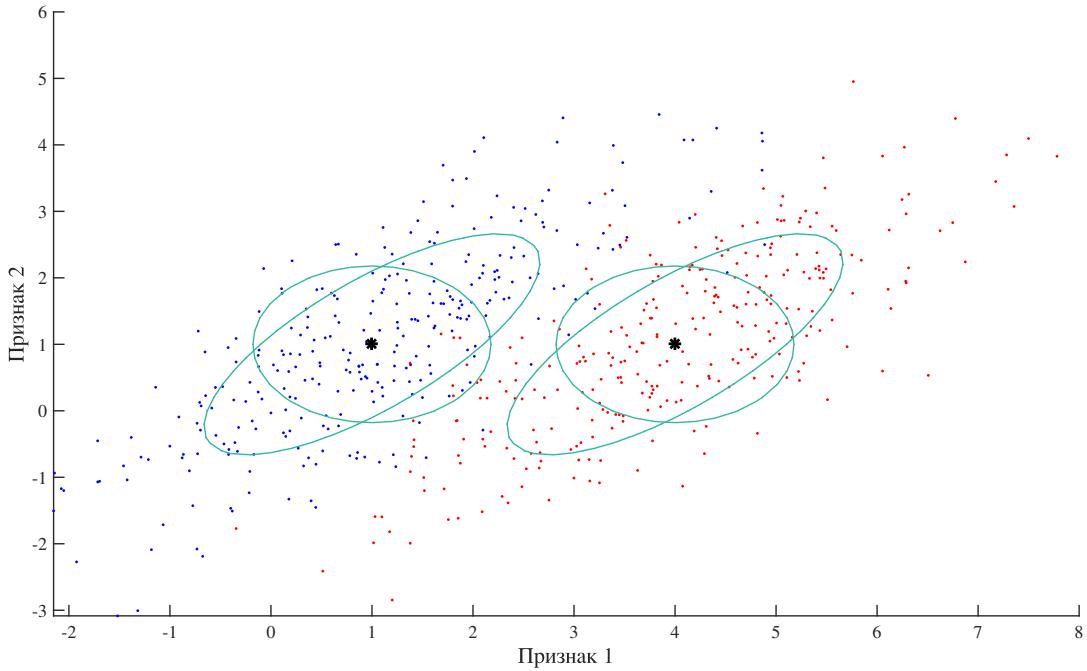


Рис. 4.1: Истинное распределение двумерных модельных данных

альных данных. Алгоритм был применен к 5 выборкам, взятых из репозитория UCI [65]. Оценкой качества кластеризации служит число правильно кластеризованных объектов. При кластеризации объектов на более чем два класса возникает проблема соотнесения истинных классов с полученными кластерами. Данная проблема была формализована в виде задачи о назначениях и решена с помощью венгерского алгоритма. Вычислительный эксперимент на реальных данных показал увеличение точности кластеризации при использовании метрического обучения.

4.5. Вычислительный эксперимент

Цель вычислительного эксперимента — проверить работоспособность предложенного подхода. Предполагается, что построенный алгоритм мультиклассовой классификации способен определить тип активности человека по форме сигнала акселерометра мобильного телефона.

Для проведения базового вычислительного эксперимента были подготов-

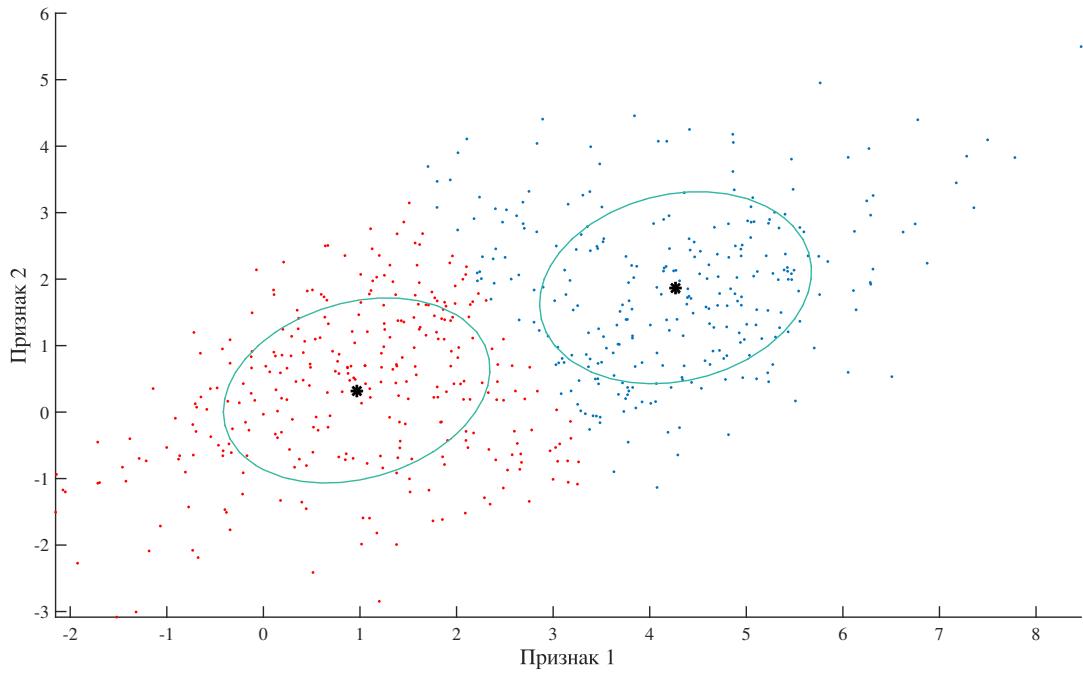


Рис. 4.2: Результат кластеризации алгоритмом k -средних

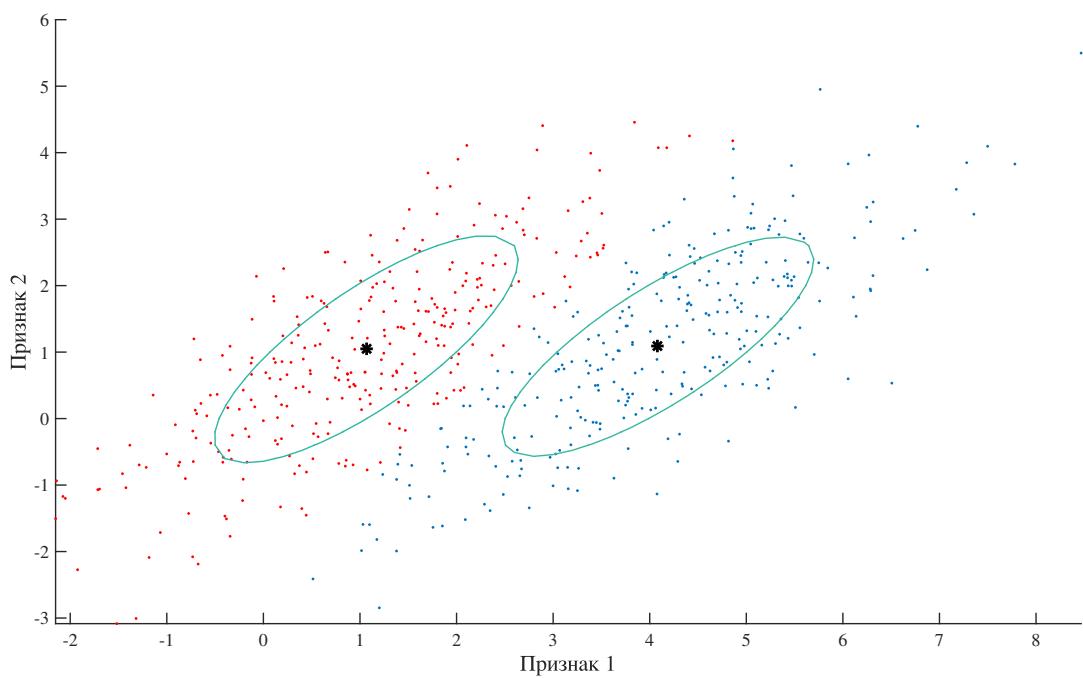


Рис. 4.3: Результат кластеризации алгоритмом адаптивного метрического обучения

Таблица 4.1: Результаты кластеризации

Выборка	Качество кластеризации	
	k -средних	AML
Letter Recognition	0,356	0,428
Optical Recognition of Handwritten Digits	0,758	0,790
Seeds	0,833	0,881
Image Segmentation	0,545	0,737
Breast Cancer Wisconsin	0,960	0,956

лены синтетические временные ряды, принадлежащие двум классам. Первый класс — синусы вида $\sin(x + b)$, где параметр b определяет сдвиг каждого временного ряда. Второй класс — пилюобразные функции с различными сдвигами по временной шкале. На каждый временной ряд был наложен нормальный шум. Число временных рядов каждого класса = 60. Длина каждого временного ряда $n = 50$.

Построенные центроиды классов проиллюстрированы на рис. 4.4. Из рисунка видно, что процедура корректно определяет сдвиги временных рядов.

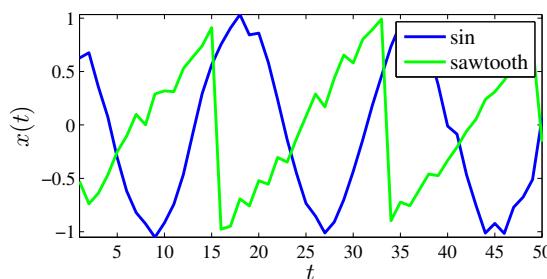


Рис. 4.4: Центроиды синтетических временных рядов

Для того чтобы убедиться в целесообразности применения метрического обучения, данные временные ряды классифицировались в пространстве с евклидовой метрикой и в пространстве с метрикой Махalanобиса. Число ближайших соседей $k = 5$, размерность преобразованного пространства $p = 40$.

Полученные ошибки классификации составили:

евклидова метрика — 27%

метрика Махalanобиса — 6%.

Реальные данные [68] представляли собой временные ряды акселерометра мобильного телефона. Каждый из шести классов соответствовал определенной физической активности испытуемых. Для проведения вычислительного эксперимента было выбрано по 200 объектов каждого класса. Длина каждого временного ряда равнялась $n = 128$ отсчетам времени.

Построенные центроиды классов изображены на рис. 4.5. Найденные центроиды обладают периодичностью, свойственной времененным рядам показаний активности человека. На рис. 4.6 показаны примеры временных рядов каждого

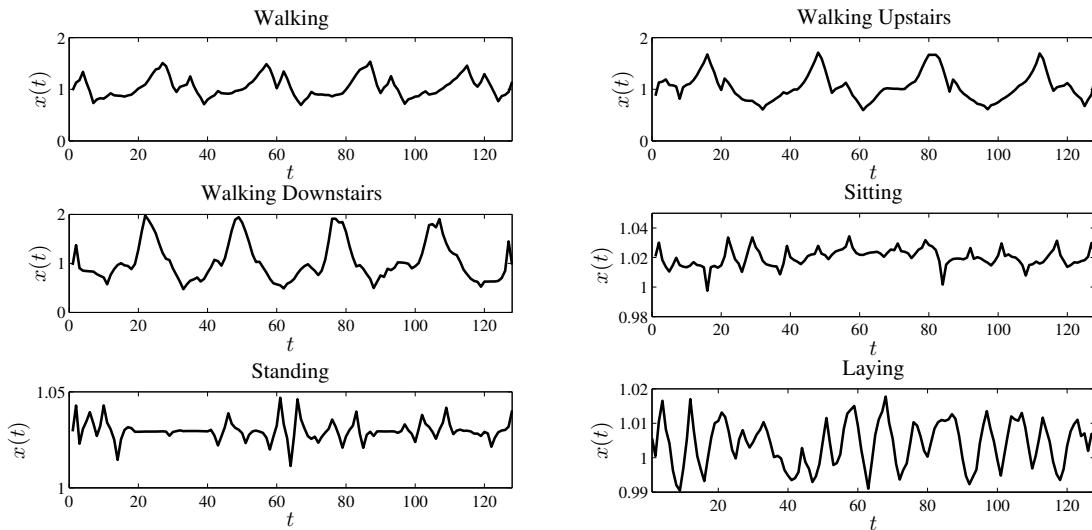


Рис. 4.5: Центроиды временных рядов акселерометра

класса. Эти же временные ряды после процедуры выравнивания относительно построенных центроидов изображены на рис. 4.7.

Ошибка классификации без использования метрического обучения составила 37,5%. Алгоритм LMNN позволяет настроить параметры: число ближайших соседей k , размерность преобразованного евклидова пространства p . Для выбора оптимальных параметров воспользуемся процедурой кросс-проверки. На

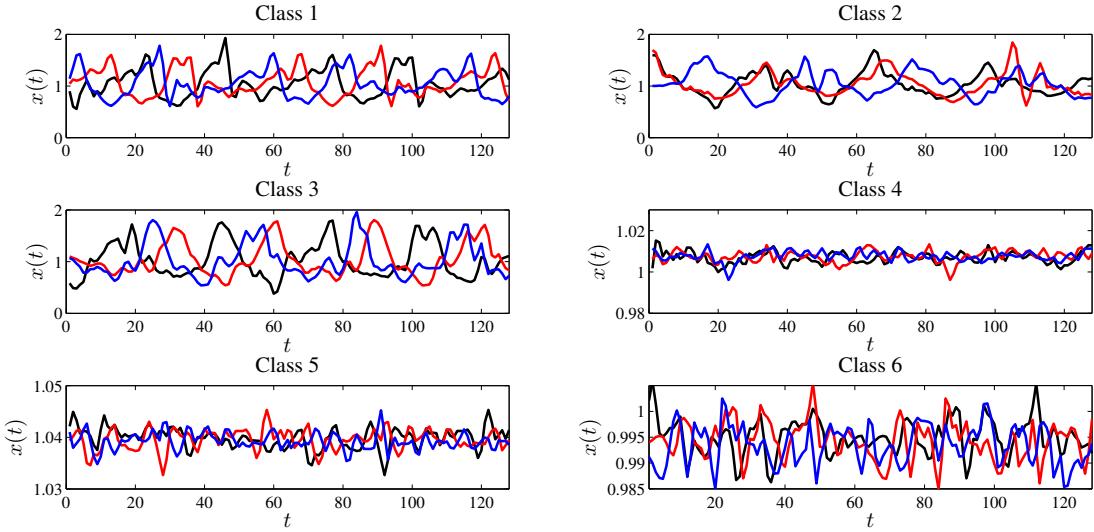


Рис. 4.6: Временные ряды акселерометра

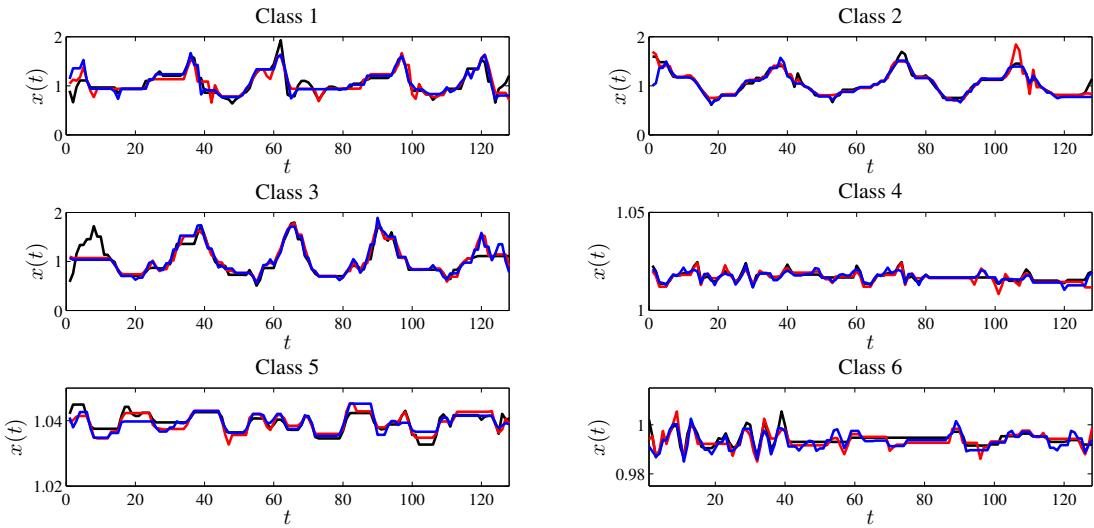


Рис. 4.7: Выравненные временные ряды акселерометра

рис. 4.8 цветом показана ошибка классификации алгоритма в зависимости от его параметров. На данной выборке алгоритм LMNN оказывается слабо чувствителен к числу ближайших соседей, и при уменьшении размерности пространства объектов ошибка классификации растет.

Настроим алгоритм LMNN со следующими параметрами: число ближайших соседей $k = 30$, размерность выходного пространства $p = 128$. Ошибка

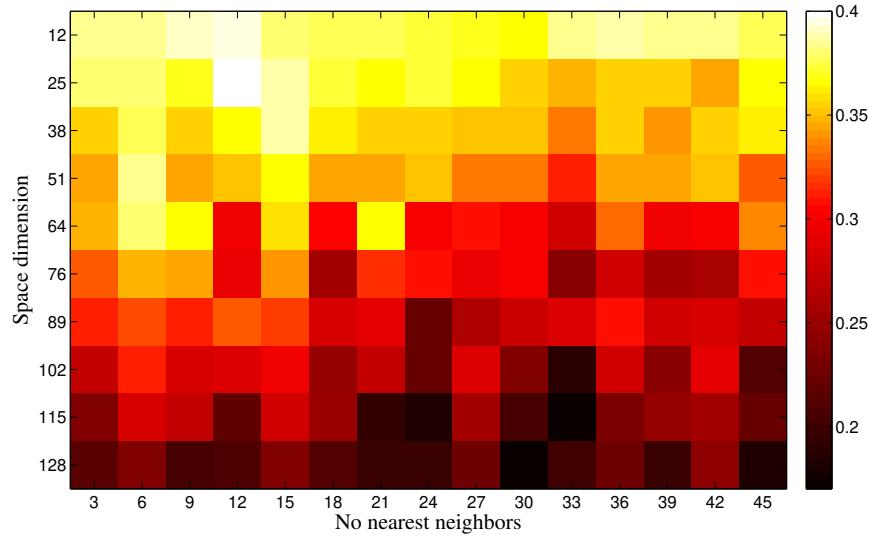


Рис. 4.8: Ошибка классификации в зависимости от параметров

классификации составила 17,25%, что вдвое меньше ошибки классификации с использованием евклидовой метрики.

Таблица 4.2: Матрицы несоответствий

(a) Евклидова метрика

	Истинные метки классов					
	1	2	3	4	5	6
1	80	0	5	0	0	0
2	4	56	33	0	0	0
3	5	5	86	0	0	0
4	7	8	5	168	4	21
5	51	61	57	12	192	11
6	53	70	14	20	2	168

(b) Метрика Махalanобиса

	Истинные метки классов					
	1	2	3	4	5	6
1	151	12	13	0	0	0
2	10	142	14	0	0	0
3	9	10	171	0	0	0
4	10	7	0	173	9	21
5	2	11	0	12	186	9
6	18	18	2	15	5	170

В табл. 4.2 представлены матрицы несоответствий результатов классификации при использовании евклидовой метрики и метрики Махalanобиса. Столбцы соответствуют истинным меткам классов объектов, строки — предсказан-

ным меткам. Диагональное преобладание матрицы несоответствий указывает на высокую предсказательную способность алгоритма.

В табл. 4.3 продемонстрировано увеличение точности классификации при использовании в качестве меры расстояния метрики Махalanобиса. Пересечение i -го столбца и j -й строки отвечает изменению доли объектов класса i , относенных к классу j . Положительное суммарное значение диагональных элементов таблицы соответствует увеличению качества классификации. Значительное улучшение предсказания происходит при классификации первых трех классов. Данные классы соответствуют следующим видам физической активности: ходьба, ходьба вверх, ходьба вниз.

Таблица 4.3: Увеличение точности классификации при использовании адекватной оценки матрицы трансформаций

	Истинные метки классов					
	1	2	3	4	5	6
1	0,355	0,06	0,04	0	0	0
2	0,03	0,43	-0,095	0	0	0
3	0,02	0,025	0,425	0	0	0
4	0,015	-0,005	-0,025	0,025	0,025	0
5	-0,245	-0,25	-0,28	0	-0,03	-0,01
6	-0,175	-0,26	-0,06	-0,025	0,005	-0,01

Глава 5

Порождение признаков с помощью метамоделей

5.1. Постановка задачи

Временные ряды акселерометра образуют множество \mathcal{S} сегментов s фиксированной длины T :

$$s = [x_1, \dots, x_T]^T \in \mathbb{R}^T. \quad (5.1)$$

Необходимо построить модель классификации $f : \mathbb{R}^T \rightarrow Y$, которая будет ставить в соответствие каждому сегменту из множества \mathcal{S} метку класса из конечного множества Y . Обозначим за

$$\mathcal{D} = \{(s_i, y_i)\}_{i=1}^m \quad (5.2)$$

исходная выборка, где $s_i \in \mathcal{S}$ и $y_i = f(s_i) \in Y$.

Авторы предлагают построить модель f в виде суперпозиции $f = f(\mathbf{g})$. Функция $\mathbf{g} : \mathbb{R}^T \rightarrow \mathbb{R}^n$ является отображением из пространства \mathbb{R}^T в признаковое пространство $G \subset \mathbb{R}^n$. Имея функцию порождения признаков \mathbf{g} , преобразуем исходную выборку (5.2) в

$$\mathcal{D}_G = \{(\mathbf{g}_i, y_i)\}_{i=1}^m,$$

где $\mathbf{g}_i = \mathbf{g}(s_i) \in G$.

Модель классификации $f = f(\mathbf{g}, \theta)$ параметризована вектором θ . Оптимальные параметры $\hat{\theta}$ определяются оптимизацией функции ошибки классификации

$$\hat{\theta} = \arg \min_{\theta} L(\theta, \mathcal{D}_G, \mu). \quad (5.3)$$

Вектор μ является внешним параметром для заданной модели классификации. Примеры таких параметров и функций ошибки для различных моделей классификации приведены ниже.

Чтобы сравнить качество классификации с прошлыми результатами [69, 70], в качестве метрики качества используется точность классификации:

$$\text{accuracy} = \frac{1}{m} \sum_{i=1}^m \left[f \left(\mathbf{g}(s_i), \hat{\theta} \right) = y_i \right]. \quad (5.4)$$

5.2. Порождение признаков

Цель данной работы — провести сравнение различных подходов к генерации признаков. В этом разделе проводится анализ рассматриваемых методов.

Экспертные функции. В качестве базового подхода будем использовать экспертные функции как функции порождения признаков. Экспертные функции — это некоторые статистики g_j , где $g_j : \mathbb{R}^T \rightarrow \mathbb{R}$. Признаком описанием $\mathbf{g}(s)$ объекта s являются значения заданных экспертных статистик для данного объекта

$$\mathbf{g}(s) = [g_1(s), \dots, g_n(s)]^\top.$$

В работе [71] авторы предлагают использовать экспертные функции, приведенные в таблице 5.1. Такая процедура порождения признаков генерирует признаковое описание временного ряда $\mathbf{g}(s) \in \mathbb{R}^{40}$.

Таблица 5.1: Expert functions

Function description	Formula
Mean	$\bar{x} = \frac{1}{T} \sum_{t=1}^T x_t$
Standard deviation	$\sqrt{\frac{1}{T} \sum_{t=1}^T (x_t - \bar{x})^2}$
Mean absolute deviation	$\frac{1}{T} \sum_{t=1}^T x_t - \bar{x} $
Distribution	Histogram values with 10 bins

Авторегрессионная модель. Авторегрессионная модель [72] порядка n использует параметрическую модель для аппроксимации временного ряда

s. Каждое значение временного ряда приближается линейной комбинацией предыдущих $n - 1$ значений

$$x_t = w_0 + \sum_{j=1}^{n-1} w_j x_{t-j} + \varepsilon_t,$$

где ε_t — регрессионные остатки. Оптимальные параметры $\hat{\mathbf{w}}$ авторегрессионной модели используются как признаки $\mathbf{g}(s)$. Данные параметры минимизируют квадратичную ошибку аппроксимации временного ряда и предсказания модели

$$\mathbf{g}(s) = \hat{\mathbf{w}} = \arg \min_{\mathbf{w} \in \mathbb{R}^n} \left(\sum_{t=n}^T \|x_t - \hat{x}_t\|^2 \right). \quad (5.5)$$

Задача (5.5) эквивалентна задаче линейной регрессии. Поэтому для каждого временного ряда s необходимо решить задачу линейной регрессии размера n . Пример аппроксимации временного ряда авторегрессионной моделью представлен на Рис. 5.1.

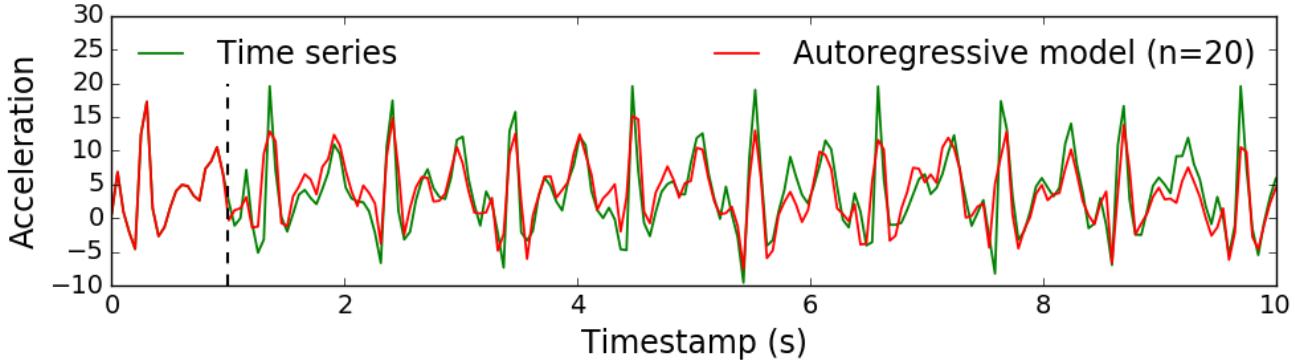


Рис. 5.1: Time series approximation using autoregressive model with order $n = 20$

Анализ сингулярного спектра. Альтернативной гипотезой порождения признакового пространства для временного ряда является анализ сингулярного спектра (Singular Spectrum Analysis, SSA) [73]. Для каждого временного

ряда s из выборки \mathcal{D} строится траекторная матрица:

$$\mathbf{X} = \begin{pmatrix} x_1 & x_2 & \dots & x_n \\ x_2 & x_3 & \dots & x_{n+1} \\ \dots & \dots & \dots & \dots \\ x_{T-n+1} & x_{T-n+2} & \dots & x_T \end{pmatrix}.$$

Здесь ширина окна n является внешним структурным параметром. Сингулярное разложение матрицы $\mathbf{X}^\top \mathbf{X}$:

$$\mathbf{X}^\top \mathbf{X} = \mathbf{U} \Lambda \mathbf{U}^\top,$$

где \mathbf{U} — унитарная матрица и $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$ причём λ_i собственные значения $\mathbf{X}^\top \mathbf{X}$. Признаковое описание объекта s задаётся спектром матрицы $\mathbf{X}^\top \mathbf{X}$:

$$\mathbf{g}(s) = [\lambda_1, \dots, \lambda_n]^\top.$$

Spline Approximation. Предлагаемый метод аппроксимирует временные ряды с помощью сплайнов [74]. Сплайн определяется его параметрами: узлами и коэффициентами. Предполагается, что узлы сплайна $\{\xi_\ell\}_{\ell=0}^M$ равномерно распределены по временной оси. Кусочные модели, построенные на отрезках $[\xi_{\ell-1}; \xi_\ell]$, заданы коэффициентами $\{\mathbf{w}_\ell\}_{\ell=1}^M$. Оптимальные параметры сплайна

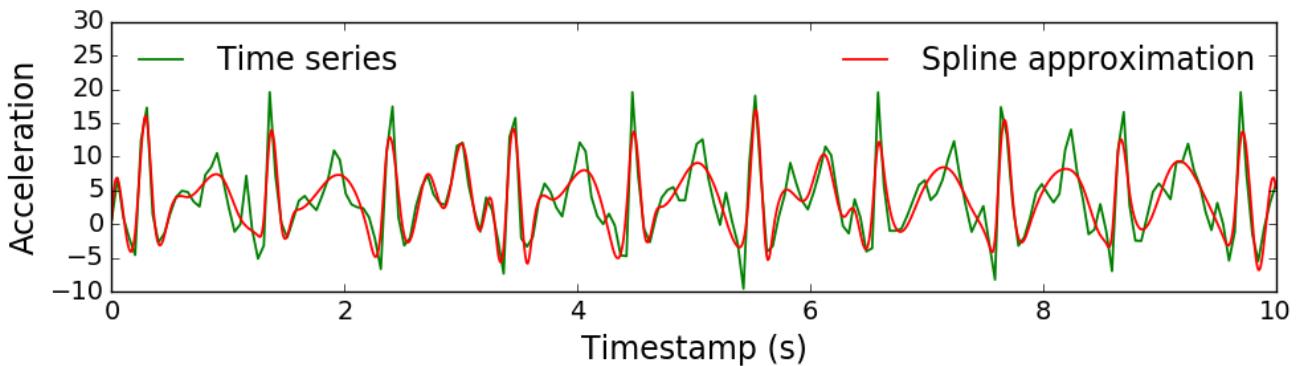


Рис. 5.2: Time series approximation using three order splines

являются решением системы с дополнительными условиями равенства произ-

водных до второго порядка включительно на концах отрезков. Обозначим каждый отрезок-сегмент $p_i(t)$ $i = 1, \dots, M$ и весь сплайн $S(t)$. Тогда система уравнений принимает вид

$$S(t) = \begin{cases} p_1(t) = w_{10} + w_{11}t + w_{12}t^2 + w_{13}t^3, & t \in [\xi_0, \xi_1], \\ p_2(t) = w_{20} + w_{21}t + w_{22}t^2 + w_{23}t^3, & t \in [\xi_1, \xi_2], \\ \dots & \dots \\ p_M(t) = w_{M0} + w_{M1}t + w_{M2}t^2 + w_{M3}t^3, & t \in [\xi_{M-1}, \xi_M], \end{cases}$$

$$\begin{aligned} S(\xi_t) &= x_t, \quad t = 0, \dots, M, \\ p'_i(\xi_i) &= p'_{i+1}(\xi_i), p''_i(\xi_i) = p''_{i+1}(\xi_i), \quad i = 1, \dots, M-1, \\ p_i(\xi_{i-1}) &= x_{i-1}, p_i(\xi_i) = x_i, \quad i = 1, \dots, M. \end{aligned}$$

Объединение всех параметров сплайна задаёт признаковое описание временного ряда:

$$\mathbf{g}(s) = [\mathbf{w}_1, \dots, \mathbf{w}_M]^\top.$$

Рис. 5.2 показывает аппроксимацию временного ряда с использованием модели сплайнов. По сравнению с авторегрессионной моделью сплайны строят более гладкую аппроксимацию, используя такое же количество параметров.

5.3. Классификация временных рядов

Для классификации временных рядов будем использовать подход один против всех. Для каждого класса обучается бинарный классификатор, и на стадии предсказания объект классифицируется согласно наиболее уверенному классификатору. Использовались три модели классификации: логистическая регрессия, SVM и случайный лес.

Логистическая регрессия. Оптимальные параметры модели $\hat{\mathbf{w}}, \hat{b}$ в случае логистической регрессии определяются минимизацией функции ошиб-

ки (5.3)

$$L(\theta, \mathcal{D}_G, \mu) = \sum_{i=1}^m \log(1 + \exp(-y_i[\mathbf{w}^\top \mathbf{g}_i + b])) + \frac{\mu}{2} \|\mathbf{w}\|^2, \text{ where } \theta = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}.$$

Решающее правило $f(\mathbf{g}, \theta)$ — знак линейной комбинации описания объекта \mathbf{g} и параметров $\hat{\theta}$

$$\hat{y} = f(\mathbf{g}, \hat{\theta}) = \operatorname{sgn}(\mathbf{g}^\top \hat{\mathbf{w}} + \hat{b}).$$

SVM. Оптимизационная задача метода SVM имеет вид

$$\hat{\theta} = \begin{pmatrix} \hat{\mathbf{w}} \\ \hat{b} \\ \hat{\xi} \end{pmatrix} = \arg \min_{\mathbf{w}, b, \xi} \frac{1}{2} \|\mathbf{w}\|^2 + \mu \sum_{i=1}^m \xi_i, \text{ s.t. } y_i (\mathbf{w}^\top \mathbf{g}_i + b) \geq 1 - \xi_i,$$

$$\xi_i \geq 0, \quad 1 \leq i \leq m.$$

Целевая функция соответствует функции ошибки классификации $L(\theta, \mathcal{D}_G, \mu)$. Предсказание для нового объекта вычисляется аналогично $\hat{y} = \operatorname{sgn}(\mathbf{g}^\top \hat{\mathbf{w}} + \hat{b})$.

Случайный лес. Случайный лес использует идею бэггинга. Идея состоит в построении многих слабых, неустойчивых классификаторов на подвыборках с возвращениями и усреднения их предсказаний. Метод предполагает использование в качестве базовых классификаторов моделей с низким смещением и высокой дисперсией. Усреднение позволяет уменьшить дисперсию. В случае случайного леса базовой моделью выступают решающие деревья. Идея бэггинга используется не только для самих объектов, но и для множества признаков. В данном случае предсказание для нового объекта получается усреднением всех предсказаний отдельных деревьев:

$$\hat{y} = \operatorname{sgn} \left(\frac{1}{B} \sum_{i=1}^B \operatorname{pred}(\mathbf{g}_i) \right),$$

где B — количество деревьев в композиции.

5.4. Вычислительный эксперимент

В данной работе эксперименты проводились на двух датасетах временных рядов акселерометра мобильного телефона: WISDM [68] и USC-HAD [75]. Акселерометр мобильного телефона проводит измерение ускорения по трём осям с частотой 100 Hz. Данные WISDM содержат 4321 временной ряд. Каждый временной ряд принадлежит к одному из 6 классов. Данные USC-HAD содержат 13620 временных рядов, принадлежащих одному из 12 классов. В таблице 5.2 представлено распределение временных рядов по классам для каждого датасета. Длина временного ряда равна 200. На рис. 5.3 представлен пример одного из временных рядов.

Таблица 5.2: Distributions of the classes

(a) WISDM			(b) USC-HAD		
	Activity	# objects		Activity	# objects
1	Standing	229 5.30 %	1	Standing	1167 8.57 %
2	Walking	1917 44.36 %	2	Elevator-up	764 5.61 %
3	Upstairs	466 10.78 %	3	Walking-forward	1874 13.76 %
4	Sitting	277 6.41 %	4	Sitting	1294 9.50 %
5	Jogging	1075 24.88 %	5	Walking-downstairs	951 6.98 %
6	Downstairs	357 8.26 %	6	Sleeping	1860 13.66 %
	Total	4321	7	Elevator-down	763 5.60 %
			8	Walking-upstairs	1018 7.47 %
			9	Jumping	495 3.63 %
			10	Walking-right	1305 9.58 %
			11	Walking-left	1280 9.40 %
			12	Running	849 6.23 %
				Total	13620

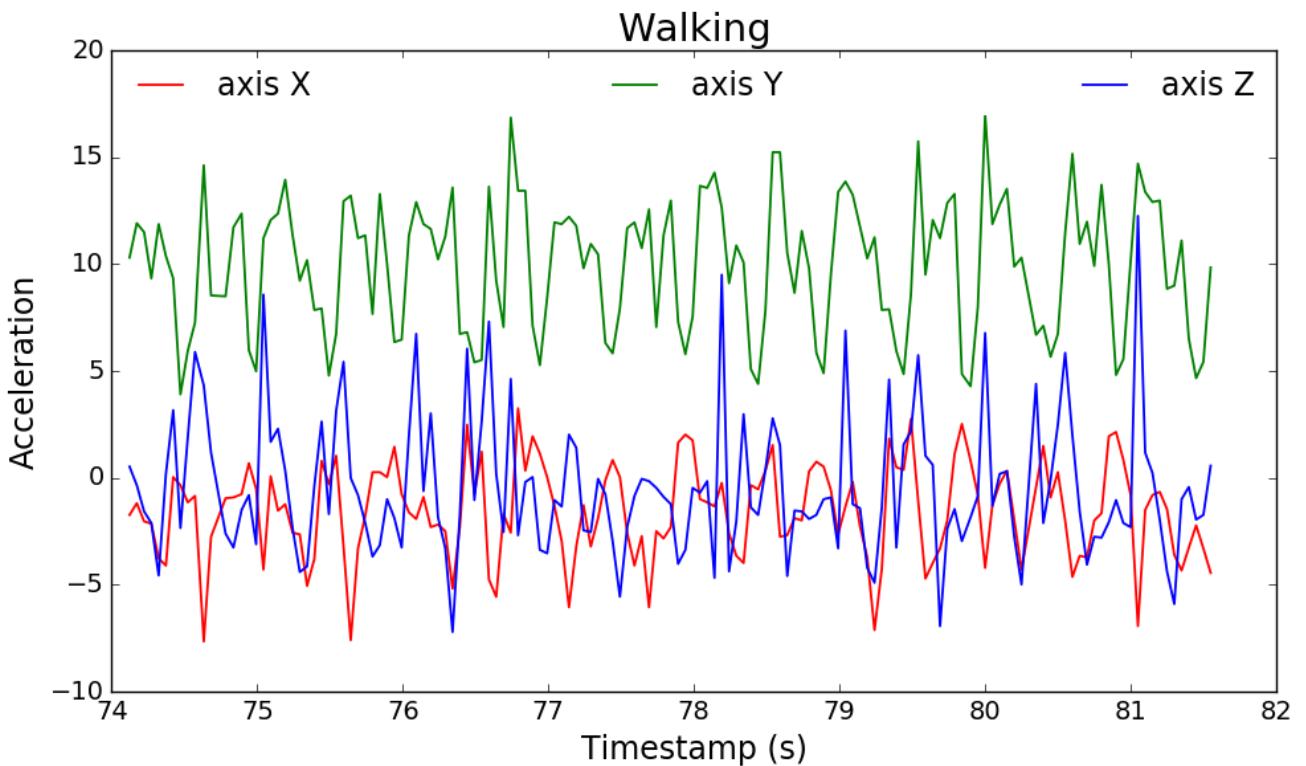


Рис. 5.3: Time series example

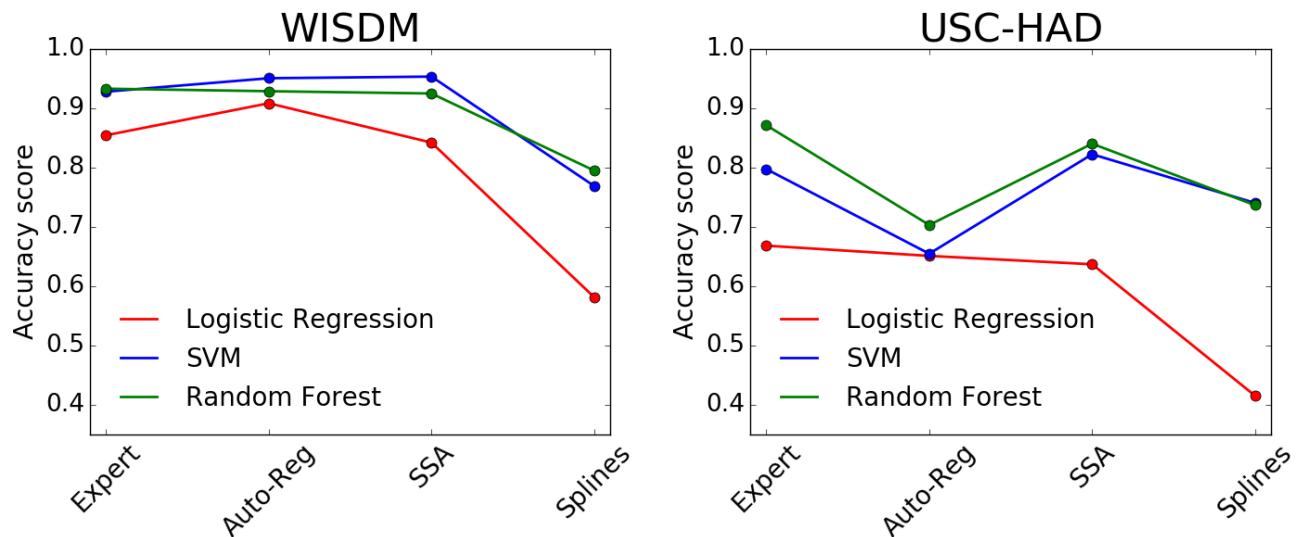


Рис. 5.4: Multiclass accuracy score

В эксперименте для каждого датасета были порождены признаки одним из методов: экспертные функции, авторегрессионная модель, SSA и сплайны. Для каждой процедуры порождения признакового описания настраивались три мо-

дели классификации: логистическая регрессия, SVM и случайный лес. Внешние структурные параметры (длина авторегрессионной модели n , ширина окна SSA n , число узлов сплайна M) настраивались процедурой кросс-валидации:

$$CV(K) = \frac{1}{K} \sum_{k=1}^K L(f_k, \mathcal{D} \setminus \mathcal{C}_k), \quad (5.6)$$

где $C_k = \frac{K-1}{K}$ доля от всей выборки, используемая для обучения модели f_k . Гиперпараметры μ моделей классификации были настроены той же процедурой кросс-валидации.

Первый подход к порождению признаков временных рядов — экспертные функции. Основной недостаток такого подхода необходимость экспертного задания функций и возможности вычисления их для конкретного датасета.

Авторегрессионная модель требует задания параметра длины модели n . Процедура кросс-валидации дала наибольшее качество при $n = 20$ для обоих датасетов.

Модель SSA была настроена аналошгичной процедурой выбора оптимальных гиперпараметров. Конечная модель имела ширину окна $n = 20$.

Для аппроксимации временных рядов кубическими сплайнами [74] использовалась библиотека *scipy*. Узлы сплайнов $\{\xi_\ell\}_{\ell=1}^M$ были распределены равномерно по временной оси. Значение параметра M было подобрано на кросс-валидации.

Для обоих датасетов процедуры порождения признаковых описаний дали следующие количества признаков: экспертные функции — 40; авторегрессионная модель — 60; анализ сингулярного спектра — 60; сплайны — 33.

На рис. 5.4 показано качество классификации (5.4) для двух датасетов. Для данных WISDM сплайны дали самое слабое качество классификации. Результаты для экспертных функций, авторегрессионной модели и SSA схожи. Для данных USC-HAD результат более восприимчив к выбору модели классификации. Для обоих датасетов логистическая регрессия продемонстрировала наименьшее качество, SVM и случайный лес показали почти одинаковое качество.

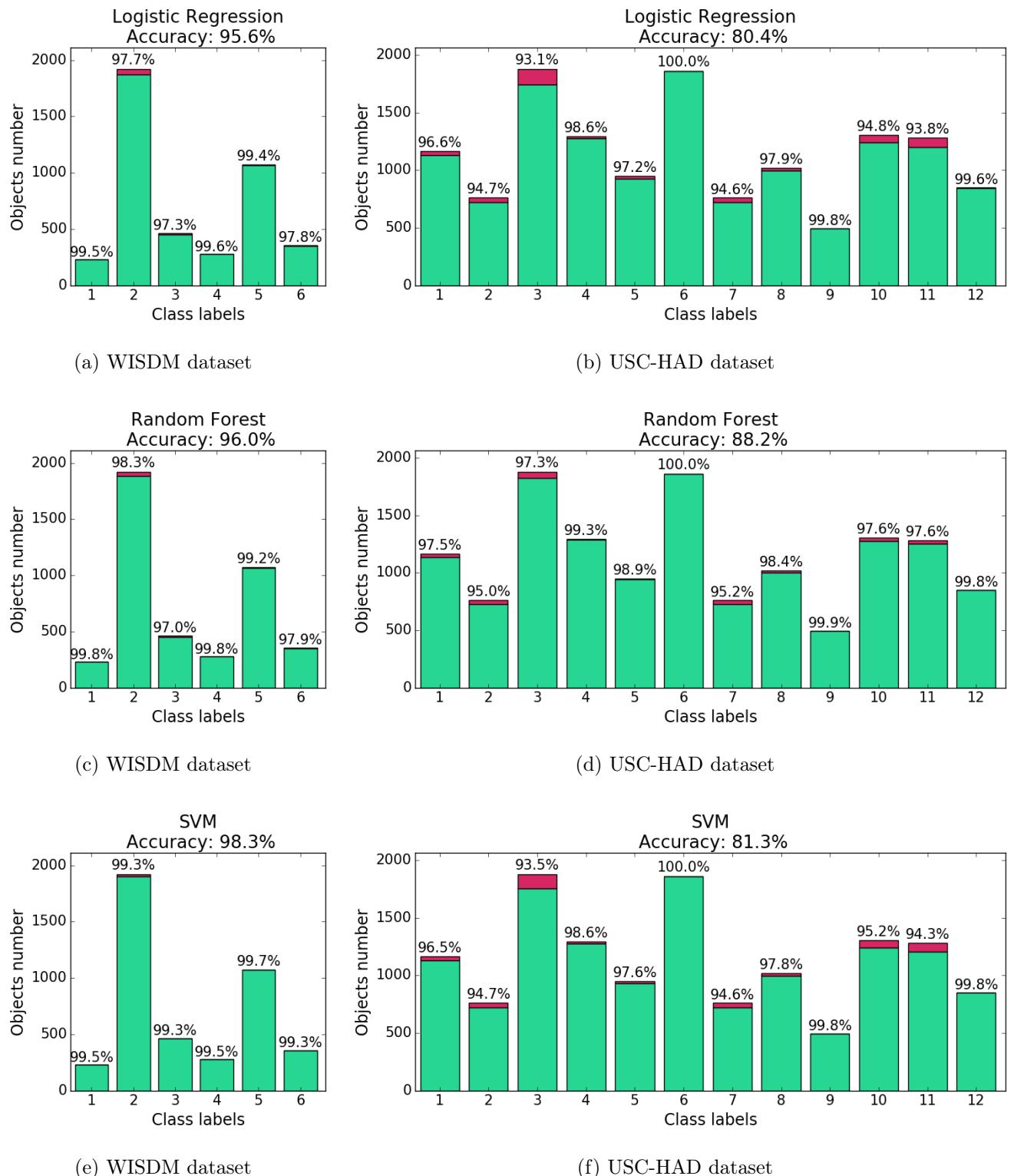


Рис. 5.5: Accuracy scores of classification of each class using all features

Для датасета USC-HAD модель с использованием аппроксимации сплайнами показала сравнимое с другими методами качество.

В Табл. 5.3 и Табл. 5.4 представлены результаты классификации (5.4) для

Таблица 5.3: Binary accuracy scores for WISDM using different feature generation methods: EX — Expert, AR — Auto-Reg, SSA and SPL for Splines

	Logistic Regression				Random Forest				SVM			
	EX	AR	SSA	SPL	EX	AR	SSA	SPL	EX	AR	SSA	SPL
All	0.85	0.91	0.84	0.58	0.93	0.93	0.92	0.79	0.93	0.95	0.95	0.77
Standing	0.99	0.98	1.00	0.95	1.00	0.99	1.00	0.99	0.99	0.98	1.00	0.96
Walking	0.91	0.96	0.86	0.61	0.96	0.97	0.95	0.86	0.96	0.98	0.98	0.84
Upstairs	0.91	0.95	0.91	0.89	0.96	0.96	0.96	0.90	0.96	0.98	0.97	0.89
Sitting	0.99	0.98	1.00	0.99	1.00	0.99	1.00	1.00	0.99	0.98	1.00	1.00
Jogging	0.98	0.99	0.99	0.80	0.99	0.99	0.99	0.92	0.99	0.99	0.99	0.93
Downstairs	0.93	0.96	0.94	0.92	0.96	0.97	0.96	0.92	0.96	0.98	0.97	0.92

Таблица 5.4: Binary accuracy scores for USC-HAD using different feature generation methods: EX — Expert, AR — Auto-Reg, SSA and SPL for Splines

	Logistic Regression				Random Forest				SVM			
	EX	AR	SSA	SPL	EX	AR	SSA	SPL	EX	AR	SSA	SPL
All	0.67	0.65	0.64	0.41	0.87	0.70	0.84	0.74	0.80	0.65	0.82	0.74
Standing	0.94	0.94	0.92	0.89	0.98	0.94	0.97	0.98	0.95	0.94	0.97	0.96
Elevator-up	0.94	0.94	0.93	0.92	0.95	0.95	0.95	0.95	0.93	0.94	0.94	0.93
Walking-forward	0.87	0.87	0.89	0.70	0.97	0.89	0.96	0.88	0.95	0.87	0.97	0.91
Sitting	0.98	0.95	0.94	0.96	0.99	0.96	0.98	0.99	0.98	0.96	0.99	0.99
Walking-downstairs	0.95	0.93	0.93	0.90	0.99	0.96	0.98	0.95	0.98	0.93	0.98	0.96
Sleeping	1.00	0.98	0.99	1.00	1.00	0.98	1.00	1.00	1.00	0.98	1.00	1.00
Elevator-down	0.94	0.94	0.94	0.91	0.95	0.95	0.95	0.95	0.93	0.94	0.94	0.93
Walking-upstairs	0.94	0.95	0.93	0.92	0.98	0.95	0.98	0.96	0.98	0.95	0.98	0.96
Jumping	0.99	0.99	1.00	0.97	1.00	0.99	1.00	0.99	1.00	0.99	0.97	0.99
Walking-right	0.91	0.90	0.91	0.86	0.97	0.92	0.96	0.92	0.96	0.90	0.97	0.93
Walking-left	0.89	0.91	0.90	0.88	0.97	0.93	0.97	0.93	0.95	0.91	0.97	0.93
Running	0.99	0.99	0.99	0.92	1.00	0.99	1.00	0.97	1.00	1.00	0.95	0.98

каждого класса в отдельности. Первая строка в обеих таблицах демонстрирует точность по всем классам для каждой модели и процедуры генерации признаков. Следующие строки соответствуют бинарным точностям по каждому из классов. Для данных WISDM лучшее качество имеют наименее активные классы, такие как Standing и Sitting. Для USC-HAD заметного выделения качества

для определенных классов не наблюдается.

Также был проведён эксперимент с использованием объединённого множества всех 193 сгенерированных признаков. Результаты представлены на Рис. 5.5. Соответствие между номером классов и видами активности приведено в Табл. 5.2. Объединение признаков для обучения одной модели позволило увеличить качество. Для данных WISDM все точности классификации по классам больше 97%, а для USC-HAD выше 93%.

Глава 6

Анализ прикладных задач

Заключение

Литература

1. Hyonho Chun and Sündüz Keleş. Sparse partial least squares regression for simultaneous dimension reduction and variable selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(1):3–25, 2010.
2. Tahir Mehmood, Kristian Hovde Liland, Lars Snipen, and Solve Sæbø. A review of variable selection methods in partial least squares regression. *Chemometrics and Intelligent Laboratory Systems*, 118:62–69, 2012.
3. A. M. Katrutsa and V. V. Strijov. Stress test procedure for feature selection algorithms. *Chemometrics and Intelligent Laboratory Systems*, 142:172–183, 2015.
4. Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM Computing Surveys (CSUR)*, 50(6):94, 2017.
5. Роман Владимирович Исаченко. Метрическое обучение в задачах мультиклассовой классификации временных рядов. In *ЛОМОНОСОВ-2016*, pages 129–131, 2016.
6. R. G. Neychev, A. P. Motrenko, R. V. Isachenko, A. S. Inyakin, and V. V. Strijov. Multimodel forecasting multiscale time series in internet of things. In *Intelligent Data Processing*, pages 130–131, 2016.
7. Роман Исаченко, Илья Жариков, and Артём Бочкарёв. Локальные модели для классификации объектов сложной структуры. In *Математические методы распознавания образов*, volume 18, pages 26–27, 2017.
8. R. V. Isachenko and V. V. Strijov. Dimensionality reduction for multicorrelated signal decoding with projections to latent space. In *Intelligent Data Processing*, pages 86–87, 2018.
9. George EP Box, Gwilym M Jenkins, and Gregory C Reinsel. *Time series analysis: forecasting and control*, volume 734. John Wiley & Sons, 2011.

10. Keith W Hipel and A Ian McLeod. *Time series modelling of water resources and environmental systems*. Elsevier, 1994.
11. John H Cochrane. Time series for macroeconomics and finance. *Manuscript, University of Chicago*, pages 1–136, 2005.
12. John W Galbraith, Victoria Zinde-Walsh, et al. Autoregression-based estimators for arfima models. Technical report, CIRANO, 2001.
13. Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.
14. Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286, 2006.
15. Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *International conference on artificial neural networks*, pages 583–588. Springer, 1997.
16. Herman Wold. Path models with latent variables: The nipals approach. In *Quantitative sociology*, pages 307–357. Elsevier, 1975.
17. Svante Wold, Arnold Ruhe, Herman Wold, and WJ Dunn, III. The collinearity problem in linear regression. the partial least squares (pls) approach to generalized inverses. *SIAM Journal on Scientific and Statistical Computing*, 5(3):735–743, 1984.
18. Herman Wold and Jean-Luc Bertholet. The pls (partial least squares) approach to multidimensional contingency tables. *Metron*, 40(1-2):303–326, 1982.
19. Paul Geladi and Bruce R Kowalski. Partial least-squares regression: a tutorial. *Analytica chimica acta*, 185:1–17, 1986.
20. Paul Geladi. Notes on the history and nature of partial least squares (pls) modelling. *Journal of Chemometrics*, 2(4):231–246, 1988.
21. Sijmen De Jong. Simpls: an alternative approach to partial least squares regression. *Chemometrics and intelligent laboratory systems*, 18(3):251–263, 1993.

22. V Esposito Vinzi, Wynne W Chin, Jörg Henseler, Huiwen Wang, et al. *Handbook of partial least squares*, volume 201. Springer, 2010.
23. Richard G Brereton and Gavin R Lloyd. Partial least squares discriminant analysis: taking the magic away. *Journal of Chemometrics*, 28(4):213–225, 2014.
24. Roman Rosipal and Nicole Kramer. Overview and recent advances in partial least squares. In *International Statistical and Optimization Perspectives Workshop Subspace, Latent Structure and Feature Selection*, pages 34–51. Springer, 2005.
25. Roman Rosipal. Nonlinear partial least squares an overview. In *Chemoinformatics and advanced machine learning perspectives: complex computational methods and collaborative techniques*, pages 169–189. IGI Global, 2011.
26. Paul Geladi. Notes on the history and nature of partial least squares (PLS) modelling. *Journal of Chemometrics*, 2(January):231–246, 1988.
27. Harold Hotelling. Relations between two sets of variates. In *Breakthroughs in statistics*, pages 162–190. Springer, 1992.
28. Theodore Wilbur Anderson. An introduction to multivariate statistical analysis. Technical report, Wiley New York, 1962.
29. Shotaro Akaho. A kernel method for canonical correlation analysis. *arXiv preprint cs/0609071*, 2006.
30. Thomas Melzer, Michael Reiter, and Horst Bischof. Nonlinear feature extraction using generalized canonical correlation analysis. In *International Conference on Artificial Neural Networks*, pages 353–360. Springer, 2001.
31. Francis R Bach and Michael I Jordan. Kernel independent component analysis. *Journal of machine learning research*, 3(Jul):1–48, 2002.
32. David R Hardoon, Sandor Szedmak, and John Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural computation*, 16(12):2639–2664, 2004.

33. David R Hardoon, Janaina Mourao-Miranda, Michael Brammer, and John Shawe-Taylor. Unsupervised analysis of fmri data using kernel canonical correlation. *NeuroImage*, 37(4):1250–1259, 2007.
34. Alexei Vinokourov, Nello Cristianini, and John Shawe-Taylor. Inferring a semantic representation of text via cross-language correlation analysis. In *Advances in neural information processing systems*, pages 1497–1504, 2003.
35. Luca Montanarella, Maria Rosa Bassani, and Olivier Bréas. Chemometric classification of some european wines using pyrolysis mass spectrometry. *Rapid Communications in Mass Spectrometry*, 9(15):1589–1593, 1995.
36. Jean-Philippe Vert and Minoru Kanehisa. Graph-driven feature extraction from microarray data using diffusion kernels and kernel cca. In *Advances in neural information processing systems*, pages 1449–1405, 2003.
37. Aria Haghghi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. Learning bilingual lexicons from monolingual corpora. In *Proceedings of ACL-08: Hlt*, pages 771–779, 2008.
38. Paramveer Dhillon, Dean P Foster, and Lyle H Ungar. Multi-view learning of word embeddings via cca. In *Advances in neural information processing systems*, pages 199–207, 2011.
39. K Choukri and G Chollet. Adaptation of automatic speech recognizers to new speakers using canonical correlation analysis techniques. *Computer Speech & Language*, 1(2):95–107, 1986.
40. Frank Rudzicz. Adaptive kernel canonical correlation analysis for estimation of task dynamics from acoustics. In *ICASSP*, pages 4198–4201, 2010.
41. Tae-Kyun Kim, Shu-Fai Wong, and Roberto Cipolla. Tensor canonical correlation analysis for action classification. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.
42. Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *International conference on machine learning*, pages 1247–1255. PMLR, 2013.

43. Weiran Wang, Raman Arora, Karen Livescu, and Jeff Bilmes. On deep multi-view representation learning. In *International conference on machine learning*, pages 1083–1092, 2015.
44. Xiaobin Chang, Tao Xiang, and Timothy M Hospedales. Scalable and effective deep cca via soft decorrelation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1488–1497, 2018.
45. Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, and Shun-ichi Amari. *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009.
46. Qibin Zhao, Cesar F Caiafa, Danilo P Mandic, Zenas C Chao, Yasuo Nagasaka, Naotaka Fujii, Liqing Zhang, and Andrzej Cichocki. Higher order partial least squares (hopls): a generalized multilinear regression method. *IEEE transactions on pattern analysis and machine intelligence*, 35(7):1660–1673, 2012.
47. Andrey Eliseyev and Tetiana Aksanova. Recursive n-way partial least squares for brain-computer interface. *PloS one*, 8(7):e69962, 2013.
48. Andrey Eliseyev and Tetiana Aksanova. Penalized multi-way partial least squares for smooth trajectory decoding from electrocorticographic (ecog) recording. *PloS one*, 11(5):e0154878, 2016.
49. Einat Kidron, Yoav Y Schechner, and Michael Elad. Pixels that sound. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, volume 1, pages 88–95. IEEE, 2005.
50. Kamalika Chaudhuri, Sham M Kakade, Karen Livescu, and Karthik Sridharan. Multi-view clustering via canonical correlation analysis. In *Proceedings of the 26th annual international conference on machine learning*, pages 129–136, 2009.
51. Raman Arora and Karen Livescu. Kernel cca for multi-view learning of acoustic features using articulatory measurements. In *Symposium on Machine Learning in Speech and Language Processing*, 2012.

52. Richard Socher and Li Fei-Fei. Connecting modalities: Semi-supervised segmentation and annotation of images using unaligned text corpora. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 966–973. IEEE, 2010.
53. Micah Hodosh, Peter Young, and Julia Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47:853–899, 2013.
54. Sarath Chandar AP, Stanislas Lauly, Hugo Larochelle, Mitesh Khapra, Balaraman Ravindran, Vikas C Raykar, and Amrita Saha. An autoencoder approach to learning bilingual word representations. In *Advances in neural information processing systems*, pages 1853–1861, 2014.
55. Manaal Faruqui and Chris Dyer. Improving vector space word representations using multilingual correlation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 462–471, 2014.
56. Jonathan Masci, Michael M Bronstein, Alexander M Bronstein, and Jürgen Schmidhuber. Multimodal similarity-preserving hashing. *IEEE transactions on pattern analysis and machine intelligence*, 36(4):824–830, 2013.
57. Janarthanan Rajendran, Mitesh M Khapra, Sarath Chandar, and Balaraman Ravindran. Bridge correlational neural networks for multilingual multimodal representation learning. *arXiv preprint arXiv:1510.03519*, 2015.
58. Abhishek Kumar, Piyush Rai, and Hal Daume. Co-regularized multi-view spectral clustering. In *Advances in neural information processing systems*, pages 1413–1421, 2011.
59. Abhishek Sharma, Abhishek Kumar, Hal Daume, and David W Jacobs. Generalized multiview analysis: A discriminative latent space. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2160–2167. IEEE, 2012.

60. Kentaro Shimoda, Yasuo Nagasaka, Zenas C Chao, and Naotaka Fujii. Decoding continuous three-dimensional hand trajectories from epidural electrocorticographic signals in japanese macaques. *Journal of neural engineering*, 9(3):036015, 2012.
61. Zenas C Chao, Yasuo Nagasaka, and Naotaka Fujii. Long-term asynchronous decoding of arm motion using electrocorticographic signals in monkey. *Frontiers in neuroengineering*, 3:3, 2010.
62. Irene Rodriguez-Lujan, Ramon Huerta, Charles Elkan, and Carlos Santa Cruz. Quadratic programming feature selection. *Journal of Machine Learning Research*, 11(Apr):1491–1516, 2010.
63. Alexandr Katrutsa and Vadim Strijov. Comprehensive study of feature selection methods to solve multicollinearity problem according to evaluation criteria. *Expert Systems with Applications*, 76:1–11, 2017.
64. Anastasia Motrenko and Vadim Strijov. Multi-way feature selection for ecog-based brain-computer interface. *Expert Systems with Applications*, 2018.
65. Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017.
66. Chris Ding, Xiaofeng He, and Horst D Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *Proc. SIAM Data Mining Conf*, 2005.
67. John Shawe-Taylor and Nello Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.
68. The wisdm dataset. <http://www.cis.fordham.edu/wisd़/dataset.php>.
69. M.E. Karasikov and V.V. Strijov. Feature-based time-series classification. *Intelligence*, 24(1):164–181, 2016.
70. M.P. Kuznetsov and N.P. Ivkin. Time series classification algorithm using combined feature description. *Machine Learning and Data Analysis*, 1(11):1471–1483, 2015.

71. Jennifer R Kwapisz, Gary M Weiss, and Samuel A Moore. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82, 2011.
72. Yu P Lukashin. Adaptive methods of short-term forecasting of time series. *M.: Finance and statistics*, 2003.
73. Hossein Hassani. Singular spectrum analysis: methodology and comparison. *Journal of Data Science*, 5(2):239–257, 2007.
74. Carl De Boor. *A practical guide to splines*, volume 27. Springer-Verlag, 1978.
75. The usc human activity dataset. <http://www-scf.usc.edu/~mizhang/datasets.html>.